ГОСУДАРСТВЕННЫЙ КОМИТЕТ СВЯЗИ, ИНФОРМАТИЗАЦИИ И ТЕЛЕКОМУНИКАЦИОННЫХ ТЕХНОЛОГИЙ РЕСПУБЛИКИ УЗБЕКИСТАН

ТАШКЕНТСКИЙ УНИВЕРСИТЕТ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ

К защите допустить
Зав. Кафедрой «АТ»
Каримова В.А____
_____2014 г.

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА

на тему: Разработка информационной системы по планированию дня для мобильного телефона

Выпускник _____ Ли В.Р.

Руководитель _____ Алимова Ф.М.

Консультант по БЖД _____Абдуллаева С.М.

Рецензент _____ _Умаров А.В.

Ташкент – 2014

ГОСУДАРСТВЕННЫЙ КОМИТЕТ СВЯЗИ, ИНФОРМАТИЗАЦИИ И ТЕЛЕКОММУНИКАЦИОННЫХ ТЕХНОЛОГИЙ РЕСПУБЛИКИ УЗБЕКИСТАН

ТАШКЕНТСКИЙ УНИВЕРСИТЕТ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ

Факультет Компьютерная инжинирия кафедра Информационные технологии

Направление (специальность): 5521900 – «Информатика и информационные технологии»

У Т В Е Р Ж Д А Ю

Зав. кафедрой _____

« ___ »_____ 2014 г.

ЗАДАНИЕ

на выпускную квалификационную работу студента Ли Вячеслава Родионовича

(фамилия, имя, отчество)

1. Тема работы: Разработка информационной системы по планированию дня для мобильного телефона.
2. Тема утверждена приказом по университету от №_____
3. Срок сдачи законченной работы: 31.05.2014
4. Исходные данные к работе: техническая литература, интернет ресурсы
5. Содержание расчётно-пояснительной записки (перечень подлежащих к разработке вопросов): Анализ предметной области, проектирование структуры БД, проектирование архитектуры мобильного приложения, реализация информационной системы на языке программирования.
6. Перечень графического материала: таблицы, пользовательские интерфейсы, диаграммы, презентация.
7. Дата выдачи задания _____

Руководитель _____

подпись

Задание принял _____

подпись

8. Консультанты по отдельным разделам выпускной работы

Раздел	Ф.И.О руководителя	Подпись дата	
		Задание выдал	Задание получил
Основная часть БЖД	Алимова Ф.М Абдуллаева С.М.		

9. График выполнения работы

№	Наименование раздела работы	Срок выполнения	Отметка руководителя о выполнении
1	Введение		
2	Анализ мобильных операционных систем		
3	Формирование постановки задачи		
4	Реализация базы данных		
5	Реализация мобильного приложения		
6	Написание пользовательского интерфейса		
7	Подготовка презентации		
8	Написание доклада		

Выпускник _____ «_____» _____ 2014 г.

(подпись)

Руководитель _____ «_____» _____ 2014 г.

(подпись)

Данная выпускная квалификационная работа посвящена разработке под мобильную операционную систему iOS, для планирования дней пользователя. В данной работе были рассмотрены теоретические основы, проблемы тайм менеджмента как предметная область, определены проблемы и выработаны требования для реализации, приведена постановка задачи и практическая реализация системы.

This Final qualification work is devoted to developing a mobile application for operating system iOS to planning user's days. In this qualification work the theoretical foundations, the problem of time management as a subject area, identified problems and developed requirements for implementation, given the problem and the practical implementation of the system.

Ушбу битирув малакавий иши iOS уяли алоқа операцион тизими асосида фойданалувчининг кунлик ишини режалашига бағишланган. Унда назарий асослар, тайм менеджмент муамолари, ишлаб чиқиш талаблари ва муамолари, тизимга қўйиладиган талаблар ва амолий ишломмаси кўриб чиқилган.

Оглавление

ВВЕДЕНИЕ

Современные информационные технологии с их стремительно растущим потенциалом и быстро снижающимися издержками открывают большие возможности для новых форм организации труда и занятости в рамках как отдельных корпораций, так и общества в целом.

Информационные технологии проникли в повседневную жизнь. Мобильные телефоны на ряду с компьютером, на данный момент играют важную роль, как средство получения и обработки, получения информации не зависимо от сферы деятельности человека.

Главное направление совершенствования мобильных телефонов можно определить одним понятием: конвергенция технологий. Мобильные телефоны объединили в себе едва ли не всё, что можно и разумно объединять под корпусом одного устройства.

Разрабатываемая информационная система является инструментом повышения технологии управления временем пользователя, так как на данный момент управление временем является важнейшим аспектом жизни человека.

Являясь важной составляющей рынка информационных технологий, рынок производства и внедрения программных продуктов обладает существенным потенциалом для повышения эффективности национальной экономики. В мировой практике имеется немало примеров активного экономического развития стран за счет отрасли программного обеспечения. Для многих стран софтверная отрасль становится новой инновационной составляющей национальной экономики.

Нашим Президентом в недавнем докладе на заседании кабинета министров, от 18 января 2013 года, посвященным итогам социально-экономического развития страны в 2012 году и важнейшим приоритетным направлениям экономической программы на 2013 год, было отмечено, что

необходимо ускорить темпы внедрения ИКТ. «Все большее значение приобретает ускоренная реализация мер и проектов в сфере информационно-коммуникационных и телекоммуникационных технологий. Мы должны отдавать себе отчет, что без кардинального, я бы сказал взрывного продвижения по пути широко внедрения во все сферы экономики, в нашу повседневную жизнь современных информационно-коммуникационных систем трудно видеть перспективу. Нам необходимо в кратчайшее время не только устранить имеющее место отставание по многим видам оказания информационных услуг, но и выйти в разряд передовых стран с высоким уровнем внедрения информационно-коммуникационных технологий» отметил Президент. Так же было отмечено, что вновь созданному Государственному комитету связи, информатизации и телекоммуникационных технологий как главному координирующему органу в этой сфере необходимо взять под жесткий контроль реализацию принятой в прошлом году Программы дальнейшего внедрения и развития информационно-коммуникационных технологий.[1]

Актуальность вопроса также усиливается необходимостью реализации задач, вытекающих из Постановления Президента Республики Узбекистан от 21 марта 2012 года «О мерах по дальнейшему внедрению и развитию современных информационно-коммуникационных технологий», определившего целевые ориентиры развитие процессов информатизации в стране на ближайшую перспективу[2].

В данной выпускной квалификационно работе рассматриваются вопросы контроля времени и систематизации хранимой информации в информационной системе «Planner».

Цель работы заключается в разработке информационной системы по планированию дня «Planner» под мобильную операционную систему iOS.

Выпускная квалификационная работа состоит из введения, 3 глав и заключения.

Во введении обоснована актуальность работы, а также определены цели и задачи необходимые для реализации.

В первой главе приведено описание предметной области а также теоретические основы

Во-второй главе приведен процесс моделирования, а также алгоритм проектирования информационной системы, приведены этапы проектирования и руководство пользователя.

В третьей главе приведены требования по охране труда и технической безопасности.

В заключении приведены основные практические и теоретические выводы к выпускной квалификационной работе.

ГЛАВА 1. ТЕОРЕТИЧЕСКИЕ ОСНОВЫ

1.1 История и анализ развития телефонов

Отправной точкой для создания мобильных приложений стало появление на мобильном телефоне экрана. Естественно, первое программное обеспечение для телефонов представляло собой встроенные приложения, которые предназначались для выполнения конкретных функций телефона и устанавливались в устройство самими производителями.

Пожалуй, первым мобильным приложением, помимо ПО, отвечающего непосредственно за работу телефона, стала телефонная книжка - та часть программного обеспечения аппарата, которая упорядочивала контакты пользователя. Сначала в записную книжку можно было занести лишь имя и номер телефона абонента. Но постепенно в данное приложение добавлялись новые функции - помимо имени и телефона, стало возможным занесение адреса, электронной почты и других данных того или иного абонента.

С появлением возможности обмена короткими текстовыми сообщениями (SMS) в телефон добавилась еще одно приложение, позволяющее писать, редактировать, отправлять небольшие электронные тексты.

Время появления первого мобильного приложения, установленного на телефон поверх уже имеющегося программного обеспечения, можно отнести к концу 90-х годов прошлого века, когда сотовая связь стала постепенно входить в жизнь миллионов людей во всем мире. Стоит отметить, что к тому времени, производители телефонов уже четко представляли, что софт для "мобильника" - это перспективное направление, как с точки зрения разработки технологий, так и с точки зрения их отдельного коммерческого использования. Тогда в программную оболочку сотовых телефонов, помимо самых необходимых приложений, производители стали устанавливать

дополнительное ПО. Как правило, это были различные мультимедийные приложения - небольшие аркадные игры, редакторы рингтонов, калькуляторы, календари и т.д.

В то время календарь было просто календарем точнее, просто для просмотра даты или же добавления минимальной информации такой как название и саму дату. Сравнивать можно было как простым бумажным календарем, где также можно было всю информацию добавит в ручную.

К тому времени рынок мобильных устройств сотовой связи стали постепенно завоевывать смартфоны и коммуникаторы. Обладая более широкими возможностями и производительностью, они отличались от обычных мобильных телефонов наличием достаточно развитой операционной системы (Windows Mobile, Symbian OS, RIM, Android, Mac OS), которая является открытой для разработки программного обеспечения сторонними разработчиками, в отличие от программной среды обычных мобильных телефонов, которая закрыта для сторонних разработчиков. При этом стоит отметить, что установка дополнительных приложений позволяет значительно улучшить функциональность смартфонов и коммуникаторов по сравнению с обычными мобильными телефонами.

С этого момента возникает эволюция сотовых телефонов и переход на производительные смартфоны и коммуникаторы.

Достоинством информационной системы “Planner” является то что система приняла в себя все современные технологии данной системы. Дружелюбный, удобный дизайн что позволяет с удовольствием и удобством пользоваться приложением. Отличительной частью информационной системы от существующих аналогов в то что, возможность добавления пользовательской информации, и разновидность типов информации. Пользователь может добавлять:

Изображение что дает возможность пользователю наиболее быстро получить доступ к изображению, которое вам нужно в определенный момент времени, а не долго искать в галереи изображений.

Локация это точка, позиция на которой отображена позиция, позицией может быть место встречи, или здания, о положении которых должен знать пользователь. Локация дает возможность не запоминать название адрес улиц или мест, а дает визуальный доступ.

Контактная информация. Добавляя контактную информацию пользователь имеет быстрый доступ к данным контакта, возможность добавления с существующих контактов смартфона делает информационную систему удобной в использовании.

Голосвоые заметки еще один полезный инструмент системы позволяющий облегчисть труд пользователя дающий возможность записывать заметки типа голосовых.

1.2 Мобильные операционные системы

BlackBerry OS — компактная операционная система для мобильных устройств с основным набором приложений. BlackBerry OS работает на ряде устройств — смартфонах и коммуникаторах, выпускаемых компанией Research In Motion Limited (RIM).

Самым современным мобильным устройством компании RIM является модель BlackBerry Torch 9800— смартфон, оборудованный сенсорным дисплеем и имеющий аппаратную клавиатуру. Аппарат получил широкий набор коммуникационных модулей, включая 3G, Bluetooth 2.0 и GPS.На нём используется новая версия операционной системы Blackberry OS 6.0

BlackBerry OS 5.0

BlackBerry OS 5.0 была выпущена компанией RIM в конце 2009 года. Основными особенностями системы являются:

- Пометка сообщений и установка времени напоминаний на смартфоне BlackBerry;
- Просмотр вложенных папок персональных контактов и редактирование контактов. BES (BlackBerry Enterprise Server) вставит все пользовательские контакты в приложение Contacts, даже если они находятся в различных папках;
- Просмотр и использование контактов, расположенных в общих папках, и копирование их в локальный список контактов пользователя, при наличии разрешения;
- Просмотрщик файлов для доступа в общие сетевые ресурсы с возможностью открывать, добавлять и сохранять документы. Возможность просмотра информации о документе, в том числе типа файла, размера и даты;
- Отправка приглашения на встречи и записи календаря со смартфона BlackBerry;
- Возможность добавлять, удалять, перемещать и переименовывать персональные папки;
- Возможность просматривать личный список рассылки в контактах Outlook и оправлять письма по нему;

RIM также работает над решением, которое позволит письмам, пришедшим со смартфона, выглядеть также, как если бы они были отправлены с Outlook. Для того, чтобы все эти возможности стали доступны пользователям, должно быть установлено как серверное, так и клиентское ПО соответствующей версии.

BlackBerry OS 6.0

Новая версия BlackBerry OS 6.0 была представлена компанией RIM в августе 2010 года. Основными особенностями системы являются:

- Новый пользовательский интерфейс предназначенный для широкого использования Multitouch-жестов, но при этом сохраняющий возможности управления с помощью трекбола. Структура рабочего стола является чем-то средним между рабочими столами операционных систем Apple iOS и Android.
- Улучшенные мультимедийные возможности ОС.
- Улучшенные возможности для веб-серфинга. В новой ОС используется мобильный браузер на основе движка WebKit, что позволяет запускать веб-приложения, написанные на языке HTML 5.
- Упрощенный доступ к мобильным сообщениям, электронной почте и социальным сетям.

Конкурирующие продукты:

Основные конкуренты BlackBerry OS — это Nokia Symbian OS, Microsoft Windows Mobile, PalmOS, Apple iOS, и различные производные ОС Linux, например: Google Android, Palm webOS, Access Linux Platform, Nokia Maemo и OpenMoko. Некоторые производители устройств также изготавливают свою собственную операционную систему для мобильных устройств.

Windows Phone 7— операционная система Windows Mobile, разработанная Microsoft, основанная на Windows Embedded CE 6.0, вышла 11 октября 2010 года. 21 октября начались поставки первых устройств на базе новой платформы. В России телефоны с Windows Phone 7 появятся в 2011 году.

Windows Phone 7 имеет новый домашний экран: здесь больше нет статичных иконок — все они заменены на так называемые «живые элементы» (Live Tiles), которые отражают информацию в режиме реального времени без участия пользователя. Например, можно создать элемент для своего друга. Просто глядя на этот элемент, пользователь всегда будет знать обо всех новых записях в социальных сетях и опубликованных фотографиях своего знакомого.

Интерфейс «Metro» полностью пересмотрен и визуально похож на интерфейс Zune HD. Microsoft переработала начальный экран, используются «плитки», которые прокручиваются по вертикали и могут быть настроены для быстрого запуска, ссылки на контакты или управление, содержатся виджеты. Windows Phone 7 Series будет иметь более дружественный пользовательский интерфейс с технологией multi-touch.

Microsoft объявила минимальные требования к устройствам на Windows Phone 7.

Все устройства должны будут обладать:

- емкостным мультисенсорным дисплеем (распознавание 4 прикосновений одновременно). Предоставляется выбор из двух вариантов разрешения: 800x480 и 320x480;
- процессором с частотой 1 Ггц;
- 256MB оперативной и 8GB флэш памяти;
- поддержкой DirectX 9;
- GPS-приемником;
- акселерометром;
- электронным компасом;
- FM-радио;
- камерой со вспышкой и разрешением не менее 5 МП.

Кроме того, выпуская устройства на базе Windows Phone 7 Series, производители больше не смогут накладывать собственные графические оболочки — интерфейс системы должен быть один и легко узнаваться вне зависимости от марки. Тем не менее, за производителями сохраняется возможность кастомизации интерфейса, отключения некоторых функций и широкий выбор в аппаратных характеристиках, включая добавление графических ускорителей. Ранее сотрудники компании Microsoft заявляли, что ОС смартфонов, работающих под управлением Windows Mobile 6.x

можно будет обновить до Windows Mobile 7-8. Но в действительности это оказалось не так, устройства, работающие под младшими версиями ОС, нельзя будет обновить до более новой версии ОС. Это связано как раз с техническими требованиями новой версии операционной системы, описанными выше.

Особенности

В Windows Phone 7-8 не будет прежней многозадачности. Вместо неё будет использована технология Tombstoning , аналогичная Push Notifications в iPhone. Возможность слушать музыку во время веб-серфинга и других действий останется. В первой версии Windows Phone 7 не будет поддержки Adobe Flash, вместо неё Microsoft предлагает поддержку Silverlight. Стив Баллмер сказал воспринимать это как факт, однако заявил, что его компания ничего не имеет против данной технологии, просто её пока в новой системе не будет. Весь существующий софт с новой версией операционной системы несовместим. Для установки приложений используются файлы с расширением .xap.

Android — это основанная на Linux платформа для мобильных телефонов, разработанная Open Handset Alliance (OHA), инициированным Google. Она позволяет создавать Java-приложения, управляющие устройством через разработанные Google библиотеки. Также есть возможность писать приложения на Си и других языках программирования с помощью Android Native Development Kit.1.5 (Cupcake) — выпущено 30 апреля 2009 года. Среди основных улучшений появилась поддержка записи и просмотра видео в режиме камеры; поддержка Bluetooth A2DP; возможность автоматически подключаться к Bluetooth-гарнитуре.

Первым устройством, работающим под управлением Android, стал разработанный компанией HTC смартфон T-Mobile G1, презентация которого состоялась 23 сентября 2008 года. Вскоре последовали многочисленные

анонсы других производителей смартфонов о намерении выпустить устройства с Android.

В компании Google выделяют несколько основных преимуществ, отличающих устройства на базе платформы Android от аналогичных продуктов:

- Открытость - Android позволяет получить доступ к основным функциям мобильных устройств с помощью стандартных вызовов API.
- Разрушение границ - можно объединять информацию из интернета с данными телефона, например контактной информацией или данными о географическом положении, чтобы получить новые возможности.
- Равноправие приложений - для Android нет разницы между основными приложениями телефона и сторонним программным обеспечением - можно изменить даже программу для набора номера или заставку экрана.
- Быстрая и легкая разработка - в SDK есть все, что нужно для создания и запуска приложений Android, включая имитатор настоящего прибора и расширенные инструменты отладки.

Кроме того, Android обладает и другими функциональными возможностями. Так, например, для выполнения приложений используется виртуальная Java-машина Dalvik с низким потреблением памяти. Dalvik позволяет поддерживать одновременную работу нескольких приложений и открывает файлы в специальном формате dex, оптимизированном для мобильных устройств.

В Android реализована поддержка 2D/3D-графики (причем одновременно можно использовать двух- и трехмерную графику), изображений, аудио и видео.

Для хранения данных используется популярная легковесная СУБД SQLite. Доступна поддержка GSM, EDGE, 3G, Bluetooth, Wi-Fi, фото- и видеокамеры, GPS, компаса, акселерометра. В работе платформы

применяется также ряд библиотек, отвечающих за шифрование данных, чтение форматов аудио и видео, поддержку 2D и 3D-графики, шрифтов и т.д. В платформе от Google также задействована библиотека LibWebCore (WebKit), которая является движком для web-браузера Android. Стоит отметить, что данный движок используется в популярном браузере Safari от компании Apple.

Одним словом, Android - это программная платформа для мобильных устройств, которая включает в себя операционную систему, программное обеспечение промежуточного слоя (middleware), а также основные пользовательские приложения (e-mail-клиент, календарь, карты, браузер, контакты и другие).

Как видно, у платформы от Google есть целый ряд преимуществ. Однако стоить обратить внимание и на недостатки Android.

Так, например, многие эксперты отмечают, что платформа базируется на Java, поэтому преимущества и возможности операционной системы Linux на Android используются не в полной мере. Также в платформе не используется ни один из популярных графических инструментов (toolkit) и библиотек (например, Ot или GTK), что делает маловероятным появление большого числа приложений, портированных с полноценного варианта Linux для домашнего компьютера на данную платформу из-за отсутствия по умолчанию X-сервера и распространенных графических библиотек.

Кроме того, появилась информация о том, что Google будет по своему усмотрению удалять приложения на телефонах пользователей, если нарушаются условия их использования.

К недостаткам платформы можно также отнести и невозможность установки приложений на карту памяти. Данный пробел разработчиков является существенным, в особенности, если у телефона небольшой объем встроенной памяти (например, у T-Mobile G1 он составляет всего 70 Мб).

Google Android устанавливается не только на смартфоны, данная платформа подходит и для нетбуков. Так, например, Android уже стоит на ряде моделей Asus EE PC, а также портирован на нетбуки компаний MSI, Dell и Acer. Еще ряд производителей нетбуков заявили о скором выпуске устройств на базе мобильной платформы от Google.

Кроме того, появление Google Android заставило многих крупных производителей микроэлектроники начать разработку устройств, которые до этого компании вообще не производили.

Аналитики и эксперты ИТ-рынка прочат Google Android хорошие коммерческие перспективы, что в принципе для продуктов на базе ПО с открытым кодом уже не является сенсацией. Они постепенно захватывают ИТ-пространство, вытесняя с него общепризнанных лидеров, порождая конкуренцию, что само по себе может только положительно сказаться на оздоровлении рынка.

iOS (до 24 июня 2010 года — iPhone OS) — мобильная операционная система, разрабатываемая и выпускаемая американской компанией Apple. Была выпущена в 2007 году; первоначально — для iPhone и iPod touch, позже — для таких устройств, как iPad и Apple TV. В отличие от Windows Phone и Google Android, выпускается только для устройств, производимых фирмой Apple.

Пользовательский интерфейс iOS основан на концепции прямого манипулирования с использованием жестов мультитач. Элементы управления интерфейсом состоят из ползунков, переключателей и кнопок.

iOS разработана на основе Mac OS X и использует тот же набор основных компонентов Darwin, совместимый со стандартом POSIX.

Для текущей версии операционной системы (iOS 6.1.3) выделяется 1,4—2 Гб флеш-памяти устройства для системного раздела

и примерно 800 Мб свободного места (варьируется в зависимости от модели).

Для хранения данных используется популярная легковесная СУБД SQLite. Доступна поддержка GSM, EDGE, 3G, Bluetooth, Wi-Fi, фото- и видеокамеры, GPS, компаса, акселерометра. В работе платформы применяется также ряд библиотек, отвечающих за шифрование данных, чтение форматов аудио и видео, поддержку 2D и 3D-графики, шрифтов и т.д

Также IOS SDK дающий быстрый старт для разработки приложений под iOS. Мощный инструментом для создания является среда разработки Xcode.

По состоянию на 19 мая 2013 года магазин приложений App Store содержит более 900 тыс. приложений для iOS, которые все вместе были загружены более 50 миллионами разработчиков. Приложения могут быть разработаны с помощью Xcode для Mac, Codea для iPad, и опубликованы в App Store — онлайн-магазине, который поставляется с самим iPhone/iPod touch/iPad.

1.3 Мобильное приложение и их классификация

Мобильное приложение – это специально разработанное приложение под конкретную мобильную платформу (iOS, Android, Windows Phone). Обычно приложение разрабатывается на языке высокого уровня и компилируется в нативный код операционной системы (ОС), дающий максимальную производительность.

Классификация приложений

- Нативные
- Веб приложения
- Гибридные

Нативные приложения загружаются через магазины приложений (App Store, Google Play или его аффилиаты, магазин приложений Windows и т.д.) и устанавливаются в ПО смартфона. Важным отличием является то, что

нативные приложения разрабатываются специально под конкретную платформу (например, под iOS для iPhone, под Android для устройств под управлением ОС Android или под Windows для Windows Phone и т.д.) и требуют от разработчика специальных знаний и умений для работы в конкретной среде разработки (xCode для iPhone, eclipse для устройств на Android); более того, используется только «родные» языки программирования для написания таких приложений. Естественно, сам процесс при этом более трудоемкий.

Таким образом, нативные приложения всегда «заточены» под конкретную ОС и органично выглядят на смартфоне. Такие приложения с легкостью могут использовать все функции ПО смартфона (камера, микрофон, акселерометр, геолокация, адресная книга, плеер и т.д.), и при этом более бережно расходуют ресурсы телефона (аккумулятор, память). В зависимости от назначения приложения предполагают или не предполагают наличие интернет-соединения.

Веб-приложения не случайно называют html5-приложениями. Это, по сути, сайт, оптимизированный под смартфон. Пользовательский интерфейс создается при помощи стандартных веб-технологий. Их не нужно загружать из магазина приложений, но они могут находиться в специальных магазинах веб-приложений, которые есть у некоторых современных браузерах, например у Chrome. Веб-приложения используют для работы браузер телефона. Главной особенностью таких приложений является их кроссплатформенность — возможность работать на всех устройствах, без дополнительной адаптации.

Независимо от установленной ОС такие приложения не могут использовать ПО смартфона. Для обновления информации в приложении необходимо подключение к интернету, скорость работы ограничена возможностями интернет-соединения провайдера услуг. При желании продавать приложение вам потребуется собственная платежная система.

На самом деле, грань между веб-сайтом, оптимизированным под мобильное устройство или с адаптивной верстkой, которая способствует адекватному его отображению на любом устройстве, и веб-приложениям очень тонкая. Как разработчики, так и пользователи в некоторых случаях путаются.

Гибридные приложения сочетают в себе некоторые функции нативных и веб-приложений: кроссплатформенность и возможность использования ПО телефона. Такие приложения могут быть загружены через магазины приложений, и при этом имеют возможность независимого обновления информации. Гибридные приложения требуют подключения к интернету, поскольку веб часть обновляется через интернет. Это, наверное, самый популярный способ построения мобильных приложений, так как у него органическая среда распространения, но разработка происходит быстрее и дешевле, чем в случае с нативными приложениями, так как, хотя оболочка и написана на «родно» языке программирования, «начинка» может быть написана в том или ином объеме на html5. Пользователь же скорее всего не заметит разницу между нативным приложением и гибридным.

В таблице 1.1 указана таблица преимуществ и недостатков классификаций

Таблица 1.1

	Преимущества	Недостатки
Нативное	• Максимальная функциональность и скорость работы • Не требуется интернет-соединение для использования	• Выше стоимость и длиннее сроки разработки • Требует от разработчика знаний

	• Имеет доступ к ПО смартфона (GPS, плеер, камера) Распространение через магазины приложений	определенной среды программирования • Работает только с одной платформой • При косметических изменениях необходимо выпускать обновление
Веб (HTML5)	• Кроссплатформенность • Не требует загрузки из магазина мобильных приложений • Можно легко адаптировать обычный сайт • Легче найти веб-разработчика нежели разработчика под определенную платформу Простота создания и поддержки	• Требует подключения к интернету • Не имеет доступа к ПО смартфона • Не может отправлять push-уведомления • Должен быть запущен интернет-браузер • При продаже требуется использование своей платежной

		системы
Гибридное	• Функциональность нативного приложения на независимой платформе • Запускается не из браузера в отличии от веб приложения Возможность независимого обновления • Распространение через магазины приложений	• Загружается из магазина мобильных приложений (необходимо соответствовать требованиям) • Разработчик должен быть знаком с разными API

1.4 Тайм менеджмент

По странному стереотипу на практике под тайм менеджментом понимают лишь небольшую и далеко не основную его часть, а именно: хронометраж времени и строгое планирование. Хронометраж — занятие довольно утомительное, отвлекающее от работы, и нужное, по большому счёту, лишь для анализа стартовых условий перехода к управлению временем, недели на две — три, да и то, в основном, для неспособных самостоятельно проанализировать свою деятельность. Строгое планирование хорошо подходит для западной культурной традиции, так называемой моноактивной культуры: задачи решаются последовательно, в единицу времени — одна задача. Но крайне сложно приживается в полиактивных культурах, к коим относимся мы, россияне. В результате очень скоро затраты, к примеру, на поддержание плана актуальным

становятся сопоставимыми с затратами на основную деятельность и начинают больше мешать, чем помогать.

Основная идея тайм менеджмента

Идей управлением рабочего времени было организация рабочего времени таким образом, чтобы эффективность его использования повышалась. Существуют общие принципы, или этапы, через которые осуществляется управление временем:

- Постановка цели. Определение и формулирование цели (целей).
- Планирование и расстановка приоритетов. Разработка плана достижения поставленных целей и выделение приоритетных (первостепенных) задач для выполнения.
- Реализация — конкретные шаги и действия в соответствии с намеченным планом и порядком достижения цели.

Контроль достижения цели и выполнения планов.

А вот каким именно способом происходит планирование или контроль результатов — тут уж процесс творческий, в котором необходимо учитывать конкретные условия и существующие возможности.

Инструментарий тайм менеджмента

Ну и пару слов о выборе инструмента. Инструмент может быть любым, главное чтоб он был удобным, и позволял сокращать затраты времени на управление, а не увеличивать их. И главное при любом инструменте — это голова руководителя. Никакая техника управления временем не принесёт результата, если нет адекватного руководства, если нет того самого тайм-менеджера, озабоченного рациональным управлением рабочим временем. Для большей наглядности рассмотрим конкретный пример, как без специальных знаний можно начать управлять временем, используя лишь здравый смысл, имеющееся в наличии программное обеспечение и базовые представления о тайм менеджменте.

Информационная система “Planner” как инструмент тайм менеджмента

Информационная система “Planner” является инструментом тайм менеджмента с помощью информационной системы решаются проблемы планирования вашего времени. Информационная система являясь приложением для мобильного телефона, что позволяет свободный доступ к планам или делам в любое время что также увеличивает актуальность информационный системы. Информационная система “Planner” разработана для операционной системы iOS. Наличие удобного дизайна и дружелюбного пользовательского интерфейса и структуры приложения, что позоляет использование приложение наиболее приятным и удобным.

1.5 Постановка Задачи

Целью данной работы является создание инструмента для тайм менеджмента, а также чтоб пользователь мог планировать свою деятельность по дням и по времени, что наиболее ускороряет работу пользователя независимо от сферы его детялеьности. Также информационная система обеспечивает удобный доступ и целостное хранение данны.

Информационная система «Planner» должен соответствовать следующим функциональным требованиям.

- Добавление заметок с установленной датой пользователя для и добавленной информации.
- Удаление заметок с определенной датой а также сохраненной в заметке пользовательской информации.
- Возможность установление приоритета заметок для установления и установки приоритетов, и сортировки заметок.
- Установления напоминаний для оповещения пользователя при неактивном приложении.
- Импорт заметок с календаря текущего устройства для плавного перехода в использовании.

- Поиск заметок для обеспечения быстрого доступа к прошедшим дням.
- Настройка цвета приоритета для удобного визуального представления.
- Установка лимита видимых дней для удобного пользования.

Вывод к первой главе

В первой главе рассмотрены история развития телефонов и анализ истории развитии телефонов. В анализе были рассмотрены проблемы планирования в телефонах в разных этапов развития. Рассмотрены текущие мобильные операционные системы их характеристики и возможности, а также инструменты для разработки. Было приведено определение мобильному приложению, приведены типы мобильных приложений и их достоинства и недостатки. Также был рассмотрено определение тайм менеджмента, основная идея и инструменты тайм менеджмента а также информационная система «Planner» как инструмент тайм менеджмента .
Дана постановка задачи и, требующие проектирования и разработки.

ГЛАВА 2. ПРОЕКТИРОВАНИЕ СИСТЕМЫ

2.1 Архитектура системы

Архитектура информационной системы построена на концепции проектирования MVC (рис 2.1).

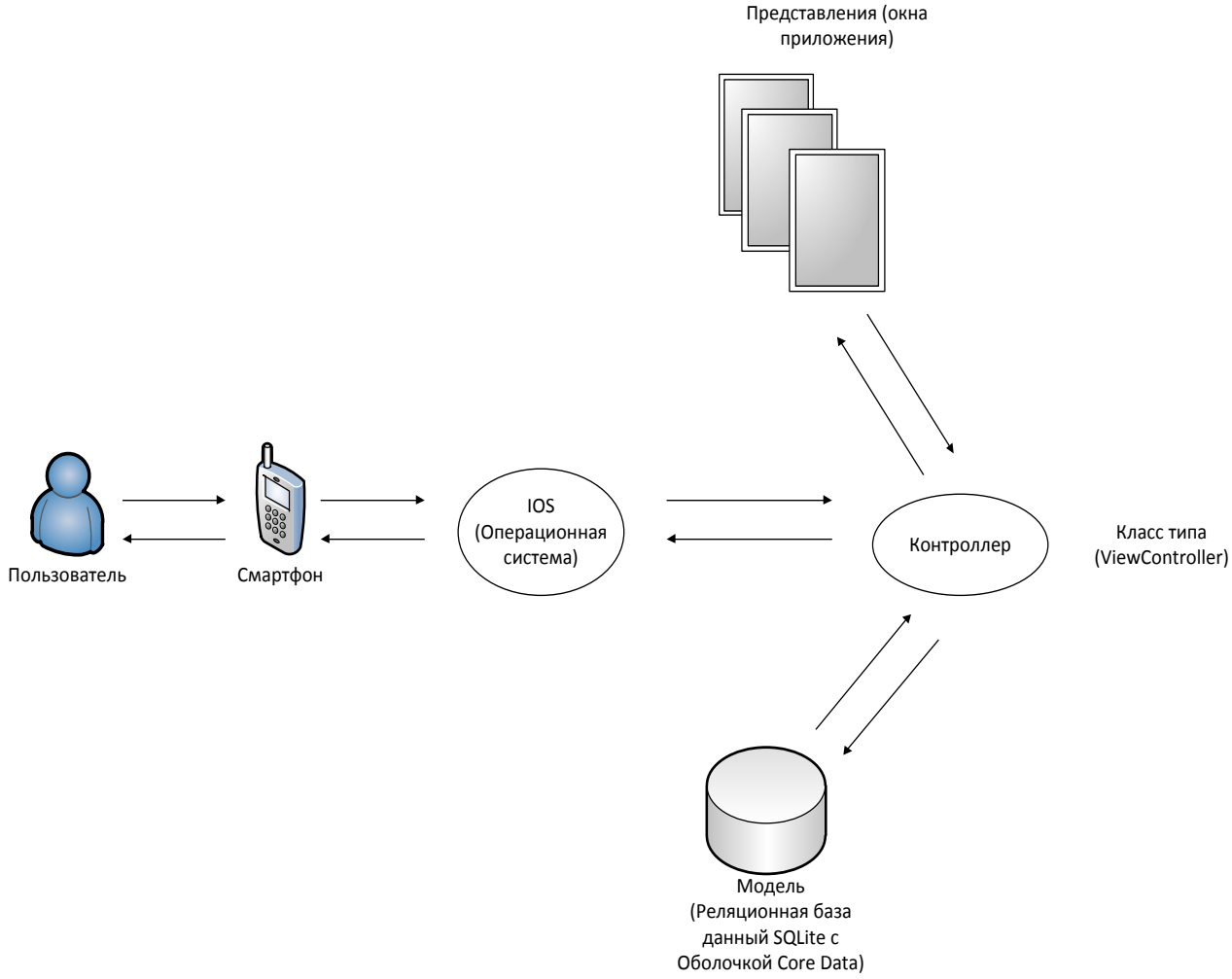

Рисунок 2.1. Архитектура системы

Model-view-controller (MVC,«модель-представление-поведение», «модель-представление-контроллер», «модель-вид-контроллер») — схема использования нескольких шаблонов проектирования, с помощью которых модель данных приложения, пользовательский интерфейс и взаимодействие с пользователем разделены на три отдельных компонента таким образом, чтобы модификация одного из компонентов оказывала минимальное

воздействие на остальные. Данная схема проектирования часто используется для построения архитектурного каркаса, когда переходят от теории к реализации в конкретной предметной области.

Концепция MVC позволяет разделить данные, представление и обработку действий пользователя на три отдельных компонента:

- Модель (англ. Model). Модель предоставляет знания: данные и методы работы с этими данными, реагирует на запросы, изменяя своё состояние. Не содержит информации, как эти знания можно визуализировать.
- Представление, вид (англ. View). Отвечает за отображение информации (визуализацию). Часто в качестве представления выступает форма (окно) с графическими элементами.
- Контроллер (англ. Controller). Обеспечивает связь между пользователем и системой: контролирует ввод данных пользователем и использует модель и представление для реализации необходимой реакции.

Важно отметить, что как представление, так и контроллер зависят от модели. Однако модель не зависит ни от представления, ни от контроллера. Тем самым достигается назначение такого разделения: оно позволяет строить модель независимо от визуального представления, а также создавать несколько различных представлений для одной модели.

Для реализации схемы Model-View-Controller используется достаточно большое число шаблонов проектирования (в зависимости от сложности архитектурного решения), основные из которых «наблюдатель», «стратегия», «компоновщик».

Наиболее типичная реализация отделяет вид от модели путем установления между ними протокола взаимодействия, используя аппарат событий (подписка/оповещение). При каждом изменении внутренних данных в модели она оповещает все зависящие от неё представления, и

представление обновляется. Для этого используется шаблон «наблюдатель». При обработке реакции пользователя вид выбирает, в зависимости от нужной реакции, нужный контроллер, который обеспечит ту или иную связь с моделью.

2.2 Структурная схема базы данных

Для работы приложения должна быть база данных подходящая и удобная для хранения пользовательских заметок и информации. Для информационной системы выбрана реляционная база данных SQLlite, предназначенная для мобильных операционных систем с оболочкой для управления базы данных Core Data предоставленная со стороны операционной системы iOS. Core Data является графом объектов и является гибким фреймворком Apple, в Mac OS операционных систем X и IOS. Core Data был введен в Mac OS X 10.4 Tiger и IOS с iPhone SDK 3.0.Core Data позволяет организовать реляционную модель данных сущность-атрибут, для сериализации в XML, бинарные данные или SQLite хранилище. Данные можно манипулировать с помощью объектов более высокого уровня, представляющие объекты и их связи. Core Data контролирует сериализацию версий, обеспечивая жизненный цикл и объекта управление графом объектов, в том числе сохранение. Интерфейсы Core Data взаимодействуют непосредственно с SQLite, изолируя разработчика от основной SQL. На рисунке 2.2 представлена структура базы данных приложения «Planner»..

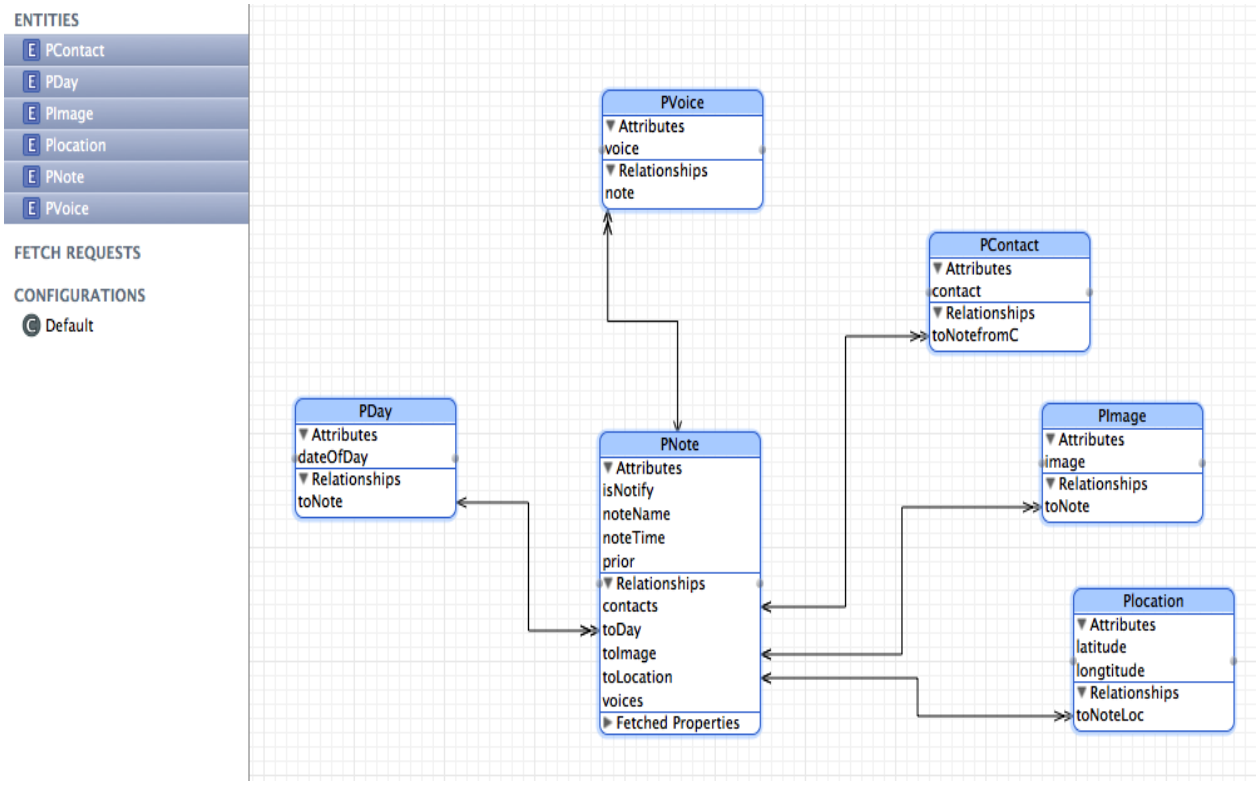

Рисунок 2.2. Структура базы данных системы.

Классы базы данных

Класс PDay – Класс дня , предназначен для хранения заметок. Свойствами класса являются дата и ссылка на класс заметок. Также автоматически сгенерированные функции добавления удаления записей с таблицы.

```
@interface PDay : NSManagedObject
@property (nonatomic, retain) NSDate * dateOfDay;
@property (nonatomic, retain) NSSet *toNote;
@end
@interface PDay (CoreDataGeneratedAccessors)
- (void)addToNoteObject:(NSManagedObject *)value;
- (void)removeToNoteObject:(NSManagedObject *)value;
- (void)addToNote:(NSSet *)values;
- (void)removeToNote:(NSSet *)values;
@end
```

```
#import "PDay.h"
@implementation PDay

@dynamic dateOfDay;
@dynamic toNote;
@end
```

Класс PNote – Класс заметок, предназначен для хранения данных заметок, такие как: изображение, локация, контактная информация, голосовые сообщения. Свойствами класса являются оповещения, имя заметки, время, приоритет и ссылки на классы Контактов, дня, изображения, голосовых заметок, локации.

```
@class PContact, PDay, PImage, PVoice, Plocation;
@interface PNote : NSManagedObject
@property (nonatomic, retain) NSNumber * isNotify;
@property (nonatomic, retain) NSString * noteName;
@property (nonatomic, retain) NSDate * noteTime;
@property (nonatomic, retain) NSNumber * prior;
@property (nonatomic, retain) NSSet *contacts;
@property (nonatomic, retain) PDay *toDay;
@property (nonatomic, retain) NSSet *toImage;
@property (nonatomic, retain) NSSet *toLocation;
@property (nonatomic, retain) NSOrderedSet *voices;
@end
@interface PNote (CoreDataGeneratedAccessors)
- (void)addContactsObject:(PContact *)value;
- (void)removeContactsObject:(PContact *)value;
- (void)addContacts:(NSSet *)values;
- (void)removeContacts:(NSSet *)values;
```

```
- (void)addToImageObject:(PImage *)value;
- (void)removeToImageObject:(PImage *)value;
- (void)addToImage:(NSSet *)values;
- (void)removeToImage:(NSSet *)values;
- (void)addToLocationObject:(Plocation *)value;
- (void)removeToLocationObject:(Plocation *)value;
- (void)addToLocation:(NSSet *)values;
- (void)removeToLocation:(NSSet *)values;
- (void)insertObject:(PVoice *)value inVoicesAtIndex:(NSUInteger)idx;
- (void)removeObjectFromVoicesAtIndex:(NSUInteger)idx;
- (void)insertVoices:(NSArray *)value atIndexes:(NSIndexSet *)indexes;
- (void)removeVoicesAtIndexes:(NSIndexSet *)indexes;
- (void)replaceObjectInVoicesAtIndex:(NSUInteger)idx withObject:(PVoice
*)value;
- (void)replaceVoicesAtIndexes:(NSIndexSet *)indexes withVoices:(NSArray
*)values;
- (void)addVoicesObject:(PVoice *)value;
- (void)removeVoicesObject:(PVoice *)value;
- (void)addVoices:(NSOrderedSet *)values;
- (void)removeVoices:(NSOrderedSet *)values;
@implementation PNote
@dynamic isNotify;
@dynamic noteName;
@dynamic noteTime;
@dynamic prior;
@dynamic contacts;
@dynamic toDay;
@dynamic toImage;
@dynamic toLocation;
```

@dynamic voices;

@end

Класс PImage – Класс изображений, предназначен для хранения картинок. Свойствами класса являются изображение и ссылка на класс заметок

@class PNote;

@interface PImage : NSManagedObject

@property (nonatomic, retain) UIImage* image;

@property (nonatomic, retain) PNote *toNote;

@end

@implementation PImage

@dynamic image;

@dynamic toNote;

@end

Класс PVoice – Класс голосовых заметок, предназначен для хранения голосовых заметок. Свойствами класс являются бинарная дата и ссылка на класс заметок.

@class PNote;

@interface PVoice : NSManagedObject

@property (nonatomic, retain) NSData* voice;

@property (nonatomic, retain) PNote *note;

@end

@implementation PVoice

@dynamic voice;

@dynamic note;

@end

Класс PLocation – Класс локации, предназначен для хранения локации. Свойствами класса являются долгота и широта.

@interface Plocation : NSManagedObject

@property (nonatomic, retain) NSNumber * latitude;

@property (nonatomic, retain) NSNumber * longtitude;
@property (nonatomic, retain) PNote *toNoteLoc;
@end
@implementation Plocation
@dynamic latitude;
@dynamic longtitude;
@dynamic toNoteLoc;
@end

Класс PContact – Класс контактной информации, предназначен для хранения контактной информации. Свойствами класса являются строка контакта и ссылка на класс заметок.

@class PNote;
@interface PContact : NSManagedObject
@property (nonatomic, retain) NSString* contact;
@property (nonatomic, retain) PNote *toNotefromC;
@end
@implementation PContact
@dynamic contact;
@dynamic toNotefromC;
@end

2.3 Функциональная схема работы мобильного приложения «Planner»

Функциональная блок схема добавление записи в таблицу PDay. Со стороны пользователя происходит запуск приложения и переход на главное окно, в этот момент просиходит поиск записи в базе данных, если в таблице дня базы данных сущетсвует запись с текущем числом, то просиходит вывод заметок текущего дня. Если не сущетвует такой записи с такой датой, то

происходит ее создание и сохранение. Аналогично происходит и при скролинге (рис 2.3).

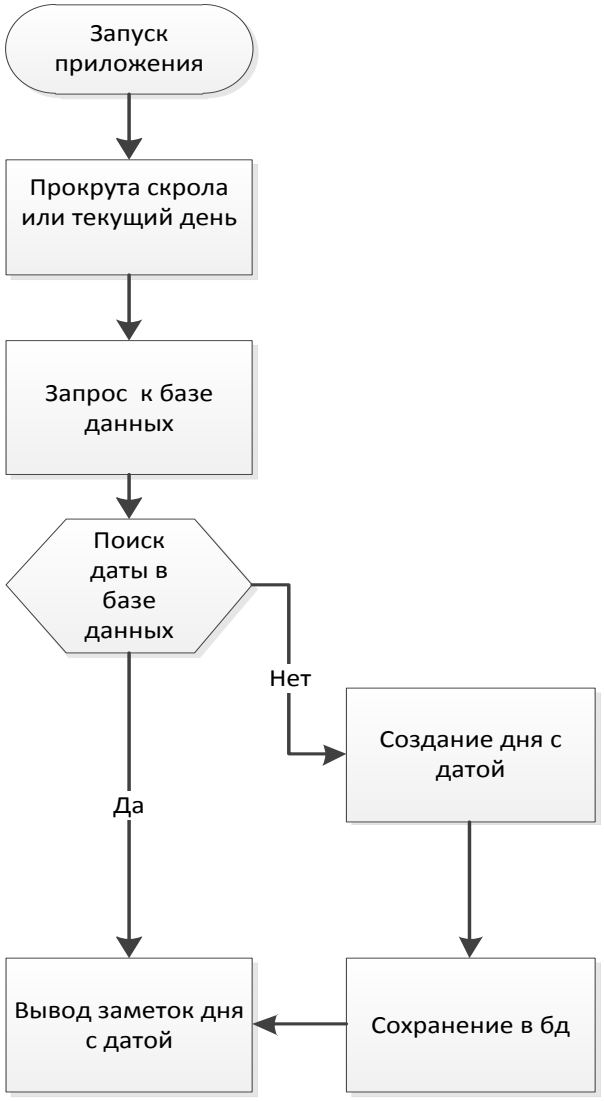

Рисунок 2.3. Схема добавления записи в таблицу PDay

Функциональная блок схема добавление записи в таблицу PNote

После перехода в окно добавениея заметки и заполнением пользовательской информаци происходит выполняется выше указанная функция. После выше указанной функции происходит добавление записи и сохранение (рис 2.4).

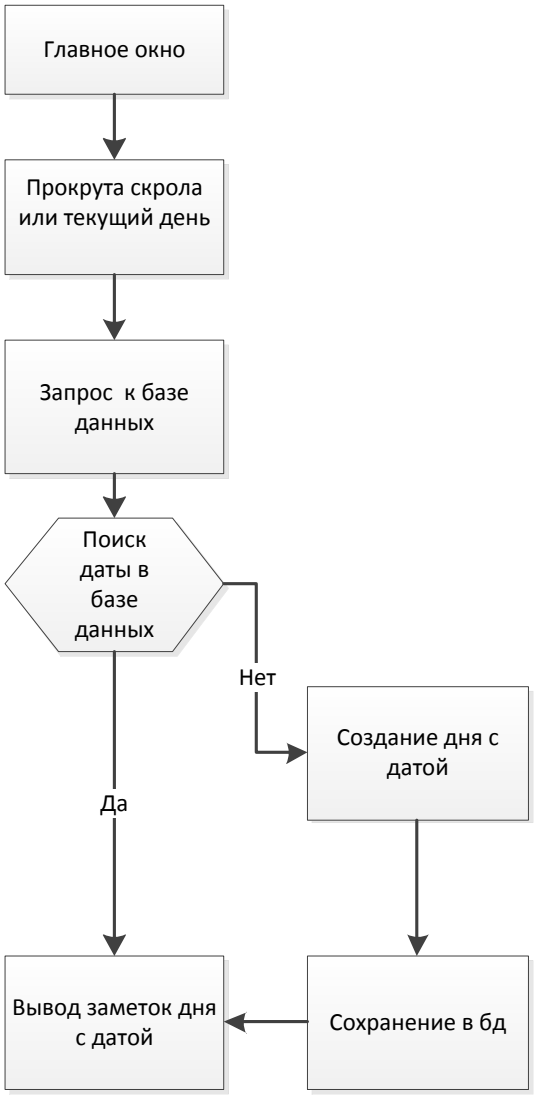

Рисунок 2.4. Схема добавления записи в таблицу PNote.

Функциональная блок схема удаление записи из таблицу PNote

В гланом окне со стороны пользователя происходит событие свайп и клик на кнопку удалить то происходит запрос на удаление такой записи из таблицы Pnote а также сохранение текущего состоянии базы данных (рис 2.5).

Рисунок 2.5. Удаление записи из таблицы PNote

2.4 Руководство пользователя

Просмотр заметок и навигация

Просмотр заметок и вложенные пользательской информации, пользователь может осущетсвить при запуске приложения или же при переходе на окно с заметками если пользователь не находится в этом окне В

главном окне находится таблица в которой хранятся заметки. В главном окне находится таблица в которой хранятся заметки (рис 2.6).

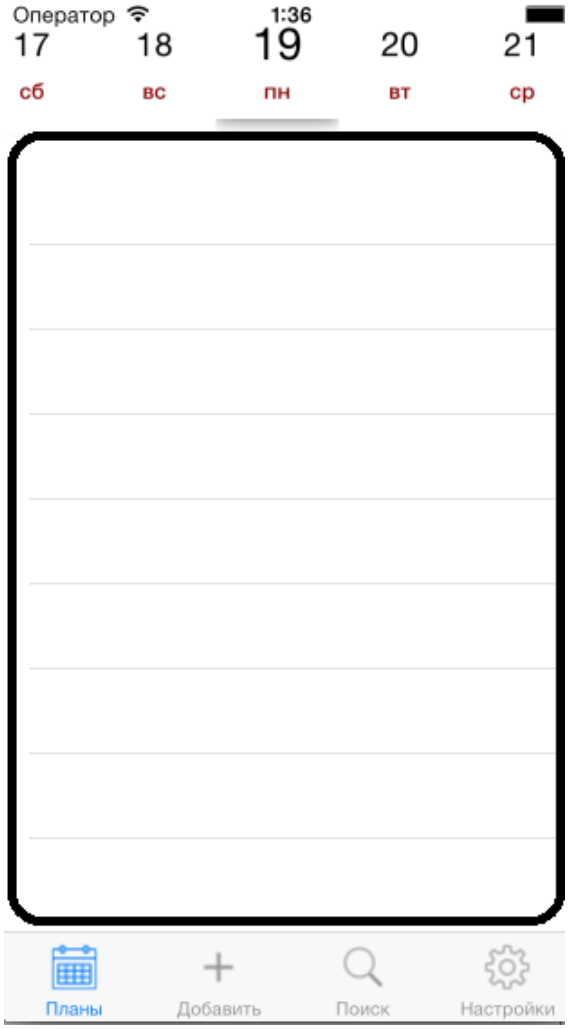

Рисунок 2.6. Главное окно

Также в главном окне находится скролл для прокрутки дней (рис 2.7).

Рисунок 2.7. Скролл дней в гланом окне дней.

Для навигации по главным окнам используется нижняя панель (рис 2.8).

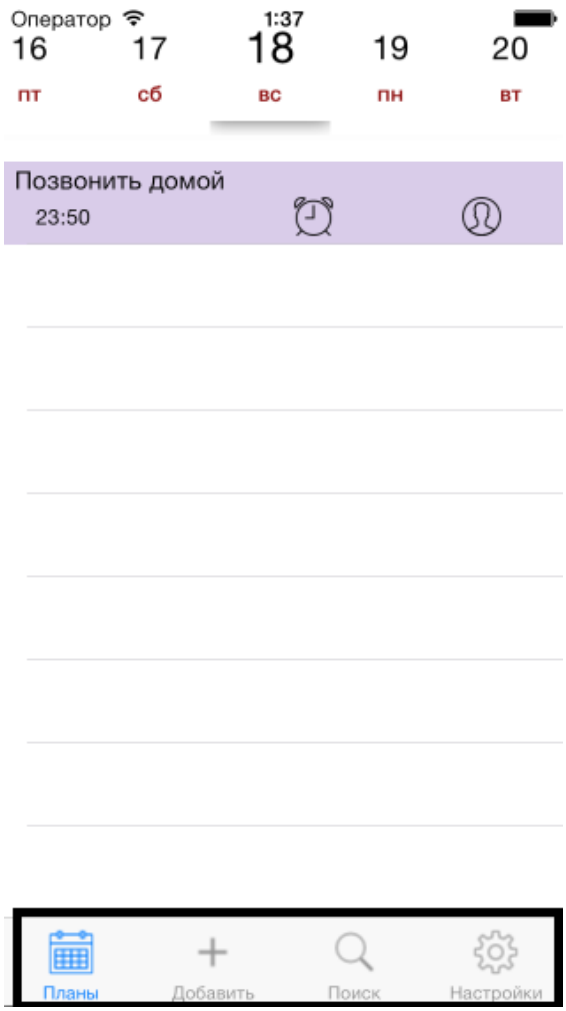

Рисунок 2.8. Панель навигации

Если в заметках показаны иконки то значит в них есть пользовательская информация (рис 2.9).

Рисунок 2.9. Заметки с информацией

Иконки кликабельны что дает возможность просмотра содержимого (рис 2.10-2.13).

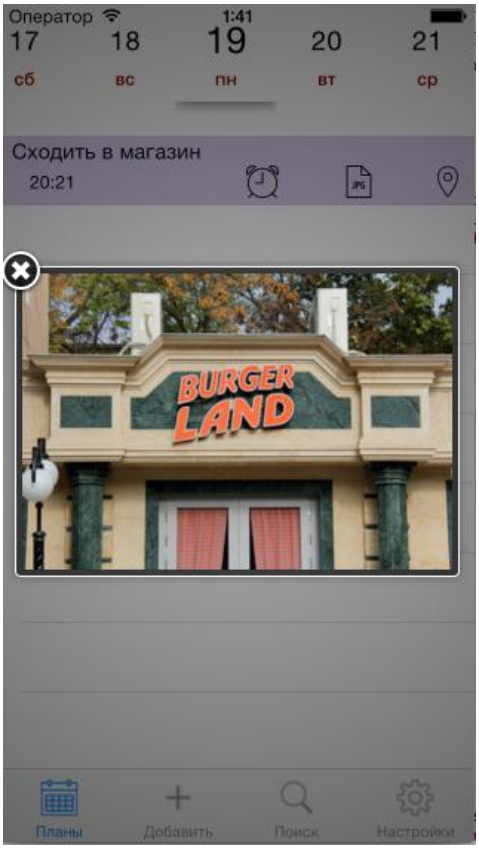

Рисунок 2.10. Изображение в заметке

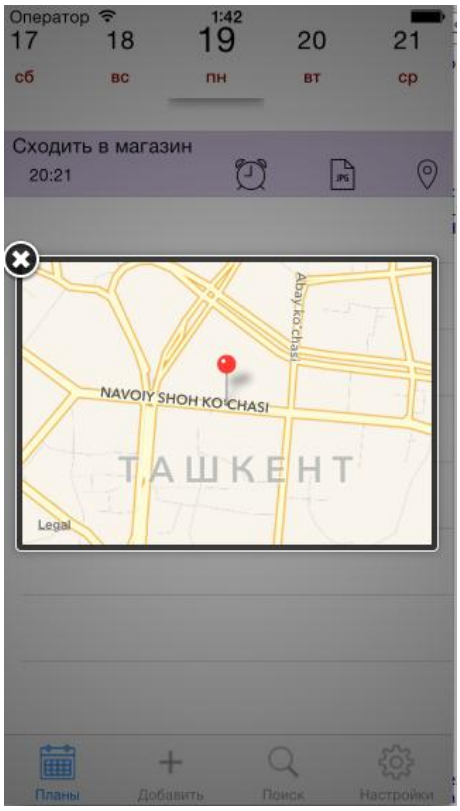

Рисунок 2.11. Отметка на карте (локация) в заметке

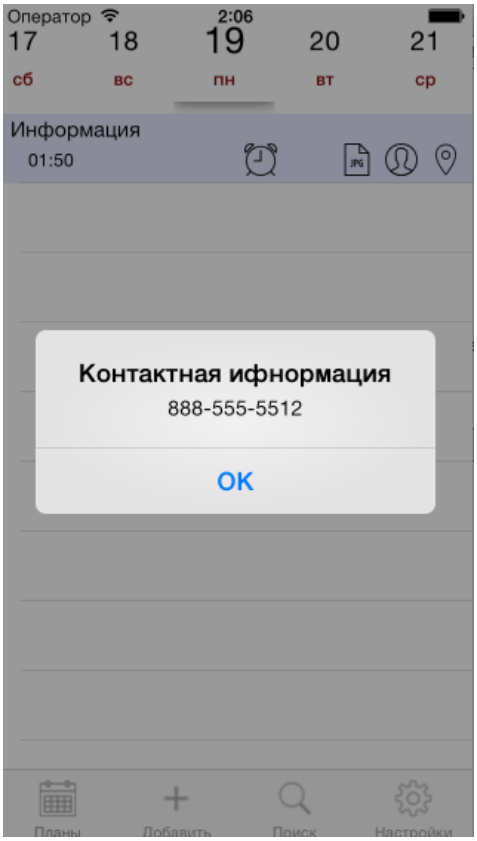

Рисунок 2.12. Отметка на картке (локация) в заметке

Добавление и удаление заметок

Окно добавления заметок является также главным экраном приложения, также следует учитывать полнотудобавления информации. При добавлении заметки нужно обязательно ввести название заметки, так же есть выбор типов информации которых может содержать заметка, такие как: Изображение, Локация, Контактная информация, голосвая заметка (рис 2.13).

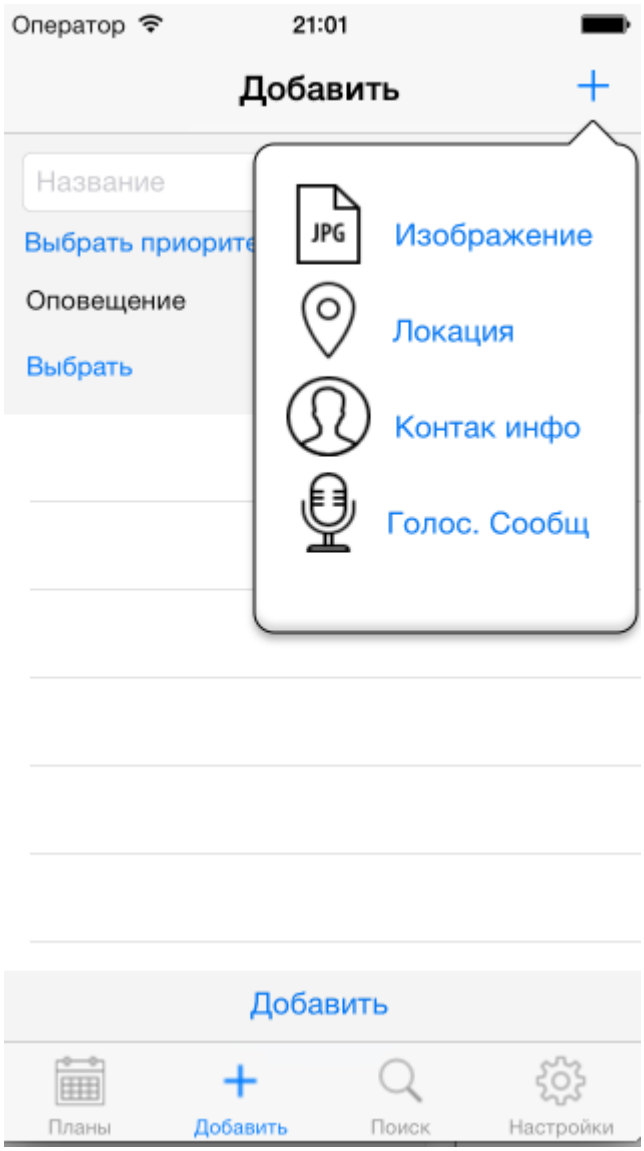

Рисунок 2.13. Выбор типа информации

В приложение существует возможность выбора приоритета что позволяет расставлять заметки по приоритету. В приложение существет три типа приоритетов: низкий, обычный, высокий. Наличеи цвета приоритета позволяет визуально различать приоритетные дела (рис 2.14).

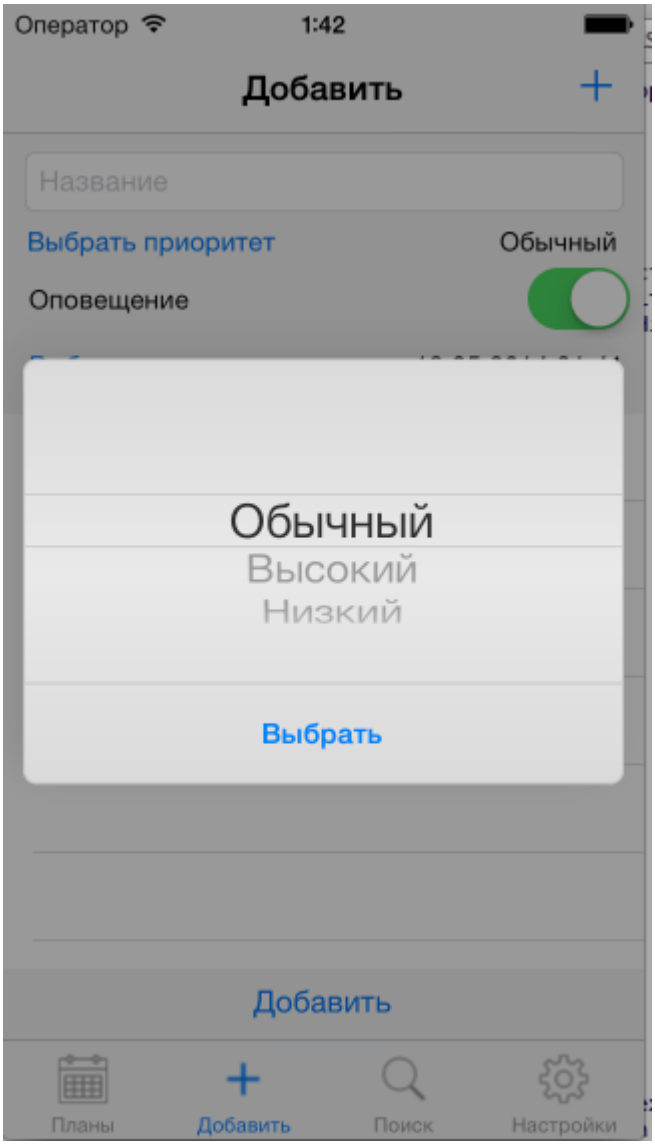

Рисунок 2.14. Выбор приоритета.

Пользователю также предоставляется возможность включения напоминай заметки что позволяет пользователяю вовремя узнавать о заметке (рис 2.15).

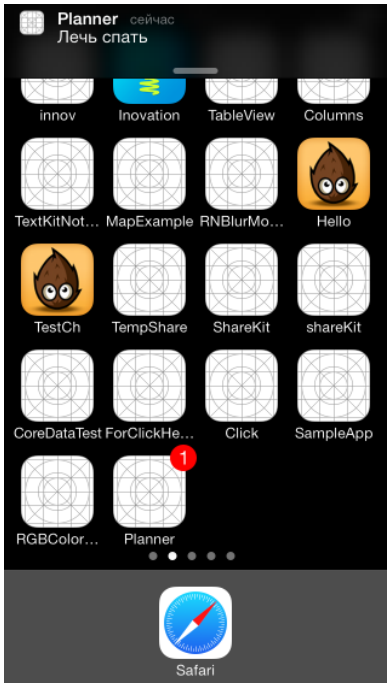

Рисунок 2.15. Напоминая в при закрытом приложении

Удаление заметок возможно из главного экрана дней при правом свайпе пользователь видит кнопку удалить, после клика на кнопку производится удаление заметки этого дня (рис 2.4.11).

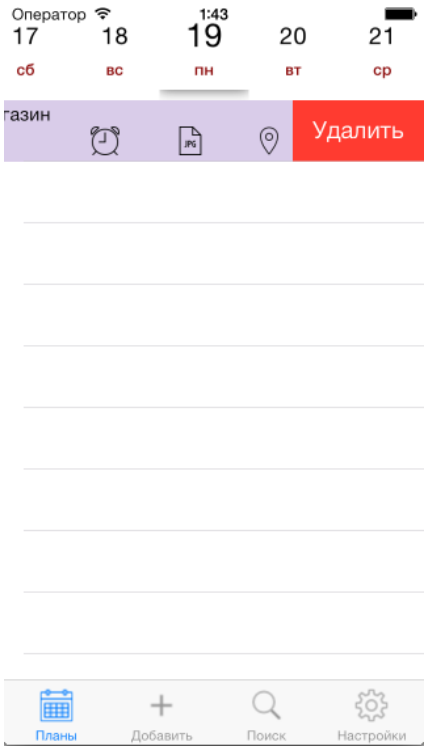

Рисунок 2.16. Удаление заметки

Поиск заметок

Поиск является также важной частью приложения и также окно поиская является одним из главных экранов приложения. Поиск доступен по трем параметра (рис 2.17).

- Имя заметки
- Дата аметки
- Приоритет заметки

Рисунок 2.17. Окно поиска

Настройки

Настройки приложения находятся в четвертом огне навигационной панели. При переходе в окно настройки пользователю дается возможность установления цветовых настроек для приоритетов дней, также есть

возможность установления лимита видимых дней, что означает скроллинг дней (рис 2.18).

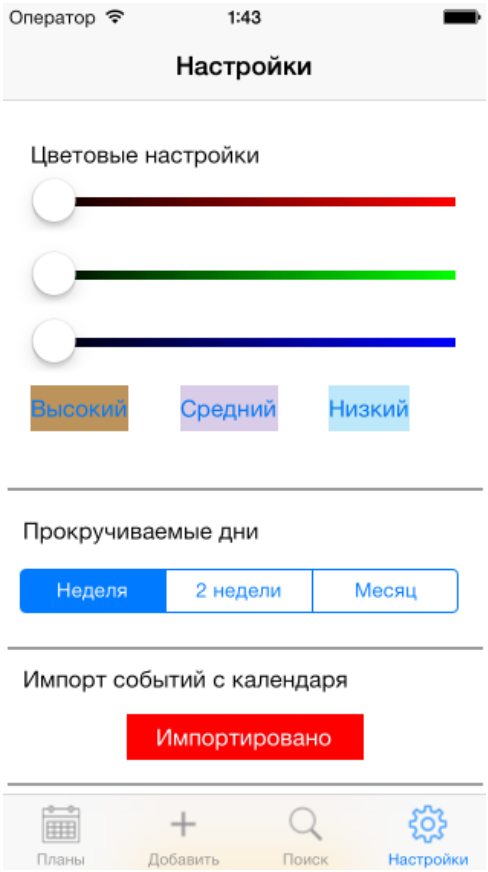

Рисунок 2.18. Окно настроек

И важной частью настроек является это импорт событий с календаря устройства, что позволяет приложению быть гибким в применении (рис 2.19).

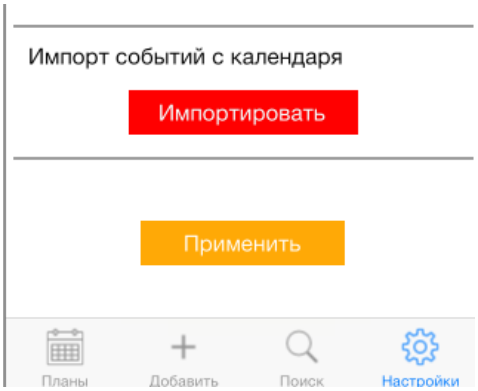

Рис 2.19. Окно настроек с кнопкой импорта

Вывод ко второй главе

В заключение второй главы приведены основные практические и теоретические:

- Рассмотрена и описана архитектура приложения.
- Рассмотрена подробная архитектура базы данных а также типа базы данных и оболочки базы данных и были представлены классы для работы с базой даных .
- Описаны основные функции и методы использованные для реализации приложения;
- Приведено подробное руководство пользователя по работе с приложением.

ГЛАВА 3. БЕЗОПАСНОСТЬ ЖИЗНЕДЕЯТЕЛЬНОСТИ

3.1 Психофизиологические нагрузки на человека

В системе мероприятий по созданию комфортных условий труда большое значение имеют рациональные режимы труда и отдыха, обеспечивающие высокую эффективность труда и сохранение здоровья работающих.

Несмотря на огромную роль человека в осуществлении производственного процесса, влияние его ограничивается психофизиологическими возможностями организма.

В основе исследования психофизиологических факторов, определяющих возможности организма, лежит понятие работоспособности человека - функционального свойства организма человека, необходимого для выполнения конкретной работы.

С физической точки зрения, это означает, что человеческий организм должен выдерживать определенные нагрузки - физическую, нервно-психическую и эмоциональную, повышать и сохранять на определенном уровне интенсивность физиологических процессов в двигательном аппарате, нервной системе, органах кровообращения, дыхательных органах и тем самым обеспечивать нормальное течение трудовой деятельности.

Выполнение любой работы в течение продолжительного времени сопровождается утомлением организма, проявляемым в снижении работоспособности человека.

Совершенствование организации и обслуживания рабочих мест неразрывно связано с улучшением условий труда, под которыми понимают совокупность элементов производственной среды, оказывающих влияние на здоровье и работоспособность человека, развитие его личности и результаты труда.

Единство психических и физических реакций работника в результате его загруженности определяет напряжение.

Напряжение следует рассматривать как нормальный биологических процесс, связанный с активностью человека. Определение степени напряжения необходимо для поддержания нормального функционирования организма. Отсутствие или недостаточность напряжения приводит к отрицательным явлениям; с другой стороны, сильное напряжение может вызвать снижение работоспособности.

Физическая нагрузка измеряется по энерго затратам. Этот метод лег в основу классификации. В зависимости от затрат физический труд делится на: тяжелый, средней тяжести и легкий физ. труд.

А) Монотонность или монотония – психическое состояние человека, вызванное однообразием восприятий или действий. Два вида монотонии: монотония за счет информационной перегрузки одних и тех же нервных центров в результате поступления большого объема одинаковых сигналов при многократном повторении единообразных движений(например, работа на конвейерах с мелкими операциями);

Монотония, вызываемая однообразием восприятия, из-за постоянства информации и недостатка новой информации(например, длительное наблюдение за приборными пультами в ожидании важного сигнала). Б) Утомление – процесс понижения работоспособности, временный упадок сил, возникающий при выполнении определенной физиологической или умственной работы.

Различают:

- быстроразвивающееся утомление(первичное) – наступает в результате выполнения работы, для которой требуются значительные физические усилия или значительное напряжение;

- медленно развивающееся утомление (вторичное) – характеризуется постепенным снижением работоспособности в результате привычной, но чрезмерно длительной и монотонной работы.

  Для предупреждения утомляемости:

- оптимальная организация режима труда и отдыха;
- рациональная организация трудового процесса;
- эффективное обучение с целью быстрого овладения трудовыми навыками.

В) Рабочая поза. Основными позами человека, представляющими интерес для производства, являются позы «стоя» и «сидя», что следует учитывать, проектируя рабочее место и рабочую позу, отвечающую данному виду работы. Необходимо стремиться к тому, чтобы рабочая поза была как можно ближе к естественной позе человека, т.к. последняя характеризуется наименьшими энергетическими затратами по сравнению с производными от них позами.

Г) Перегрузки эмоциональные и умственные. Умственная деятельность (как и мышечная) – это деятельность прежде всего центральной нервной системы, ее высшего отдела – коры человеческого мозга. При умственно работе, как и при физической, изменяются обменные процессы, но повышение общего обмена незначительно(не более 10-15%); в отличии от физической работы при умственной работе происходит сужение сосудов конечностей и расширение сосудов внутренних органов, пульс изменяется незначительно. Вместе с тем, если для умственной работы требуется значительное нервно-эмоциональное напряжение, то возможны значительные изменения кровяного давления, пульса, повышения уровня сахара в крови.

Д) Стресс – это реакция адаптации к чрезвычайным, экстремальным условиям, как физиологическим, так и психическим. Для обеспечения

безопасности труда необходимо организовывать так производственный процесс, чтобы он исключал стрессы. Необходимо, чтобы в аварийных условиях стресс не явился причиной неправильных действий и не ухудшил производственную обстановку.

Е) Гиподинамия – это нарушение функций организма(опорно-двигательного аппарата, кровообращения, дыхания) при ограничении двигательной активности, снижении сил сопротивления мышц. Профилактика гиподинамии предусматривает производственную гимнастику и т.п.

Тяжесть труда — характеристика трудового процесса, отражающая нагрузку на опорно-двигательный аппарат и функциональные системы (сердечнососудистую, дыхательную и др.).

Напряженность труда отражает значимую для человека ситуацию, является индикатором соответствия средств и условий деятельности возможностям человека, характеризуется степенью активизации функций, обеспечивающих деятельность, нервно-психическими затратами на нее и является одной из характеристик, составляющих тяжесть труда.(Рис 3.1)

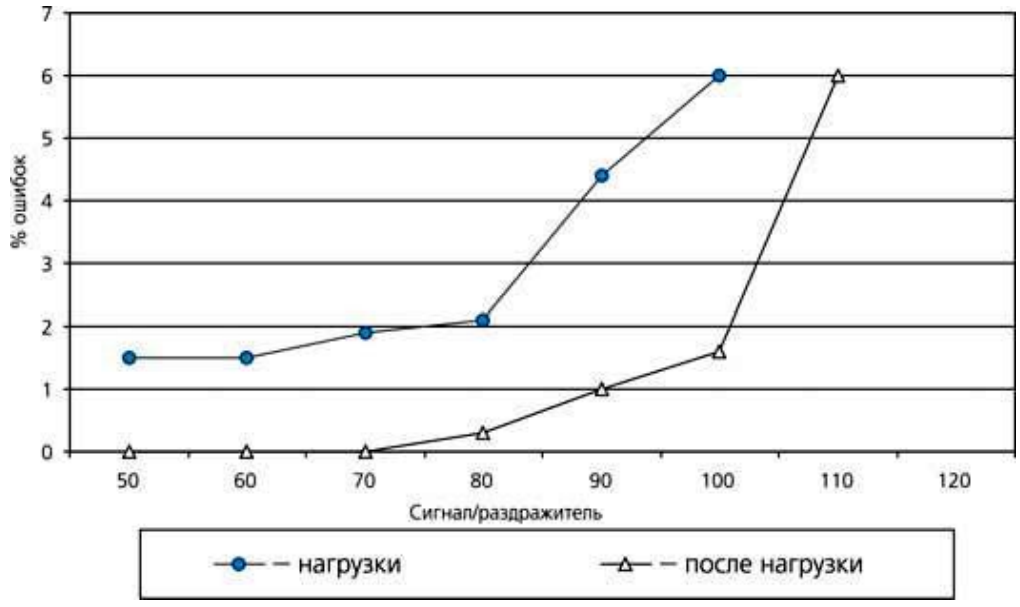

Рис. 3.1. Психофизиологические нагрузки.

Психофизиологические нагрузки приводят к перенапряжению зрительных анализаторов и возникновению нервно-эмоционального напряжения. Оптимальная психофизиологическая нагрузка обеспечивается соблюдением требований и рекомендаций, разрабатываемых инженерной психологией и эргономикой относительно объема поступающей и перерабатываемой информации, норм обслуживания оборудования и других объектов, а также путем формирования и поддержания в коллективах участков и цехов благоприятного психологического климата. Организм человека неодинаково реагирует на физическую и нервно-психическую нагрузку в разное время суток. При прочих равных условиях предпочтительнее утренние и дневные часы, которым предшествует полноценный ночной отдых и которые совпадают со временем наибольшей биологической активности.

В вечерние и особенно ночные часы физиологические процессы замедляются. Поэтому оптимальным является двухсменный режим работы предприятия. При невозможности прерывать технологический процесс, то есть при трехсменном режиме, продолжительность ночной смены должна быть меньше дневной. Работа человека слагается из мускульных усилий и психофизиологической нагрузки. Современное развитие производства с применением высокопроизводительных машин, оборудования, систем контроля и слежения за технологическими процессами, управления параметрами технологических процессов повысило долю нервно-психических нагрузок на трудящихся и требует реализации высших психических функций человека - памяти, мышления, внимания, быстрого принятия безошибочного решения, правильной реакции на любую информацию.

Во время трудовой деятельности работоспособность организма закономерно изменяется по суточному ритму. В течение суток организм по-разному реагирует на физическую и нервно-психическую нагрузку. В

соответствии с суточным циклом организма наивысшая работоспособность отмечается в утренние (с 8 до 12 ч) и дневные (с 14 до 17 ч) часы. В дневное время наименьшая работоспособность, как правило, отмечается в период между 12 и 14 ч, а в ночное время - с 3 до 4 ч, достигая своего минимума. С учетом этих закономерностей определяют сменность работы предприятий, начало и окончание работы в сменах, перерывы на отдых и сон.

При определенных условиях психофизиологические нагрузки могут быть определены как вредные и опасные производственные факторы. Эти факторы, обусловленные определенными технологическими особенностями производства, характером и организацией трудового процесса, параметрами оборудования и рабочего места, могут образовывать целый комплекс психофизиологических факторов, оказывающих отрицательное воздействие на здоровье работающего человека, ухудшающих его самочувствие, психоэмоциональное состояние и интеллектуальную деятельность, приводящих к стойкому снижению работоспособности и нарушению состояния здоровья, а также значительно повышающих риск производственного травматизма. По характеру воздействия сами психофизиологические факторы труда подразделяются на физические нагрузки – статическое и динамическое напряжение и на нервно-психические нагрузки, складывающиеся из характера умственного и эмоционального напряжения труда, напряжений анализаторов, монотонности действий и стереотипных движений при повторяющихся операциях. Все это может быть охарактеризовано определенными параметрами или показателями уровней воздействия на организм работающего человека. Соответственно этому определяются классы и степени вредности психофизиологических факторов труда как показатели их воздействия на функциональное состояние организма в целом .

Задача выявления рациональных приемов труда решается путем изучения непосредственно на производстве труда рабочих, значительно

перевыполняющих нормы труда, применяющих эффективные способы ведения трудового процесса.

При изучении и отборе наиболее целесообразных и экономных приемов труда рекомендуется обращать внимание:

- на короткие и наименее утомительные движения рук, ног, корпуса тела работающего, устранение резких перемен в направлении этих движений, уменьшении массы перемещаемых вручную грузов;
- непрерывные и плавные движения по дуговой линии, которые более экономны, чем движения прямолинейные с резкими остановками;
- одновременные и симметричные движения рук;
- сокращение движений путем исключения лишних, совмещения движений;
- достижение удобного положения рабочего, обеспечение переменной позы «сидя–стоя», чередование периодов труда и отдыха в зависимости от тяжести и нервно-психической напряженности труда.

Рациональные приемы и методы труда, опыт новаторов производства распространяются разными способами. Для этих целей используют систему подготовки и повышения квалификации кадров, семинары, выставки, учебные курсы и др.

3.2 Рациональная организация рабочего места

Рабочим местом называется определенный участок производственной площади цеха, отделения, участка или мастерской, закрепленный за данным рабочим (или бригадой рабочих) и предназначенный для выполнения определенной работы.

Каждое рабочее место оснащается комплектом организационно-технических устройств — оргтехоснасткой, которая должна обеспечить: удобства работающему при выполнении закрепленной за ним работы и безопасность труда; рациональное построение трудового процесса и

физиологически правильную рабочую позу; рациональное размещение и строгий порядок хранения инструментов, приспособлений, заготовок, готовой продукции и т. п., а также поддержание чистоты и порядка на рабочем месте.

На этом первом звене производственного процесса — рабочем месте — решаются основные производственные задачи по повышению качества продукции и производительности труда, т. е. по повышению эффективности работы предприятия. Рациональная организация рабочих мест имеет первостепенное значение в повышении рентабельности предприятий.

Научная организация труда на рабочем месте предусматривает прежде всего максимальную экономию рабочего времени. Рациональная организация рабочего места должна обеспечивать условия для высокой производительности труда; предусматривать рациональный трудовой процесс, который экономит рабочее время и силы рабочего, избавляет его от лишних и неудобных движений и обеспечивает высокую производительность труда и качество работы; максимально сокращать время на ручные приемы и др.

Для создания рациональных рабочих мест должны быть выполнены следующие требования НОТ: точно определен и закреплен состав работы на рабочем месте; установлена система обслуживания рабочих мест материалами, заготовками, инструментом, приспособлениями и деталями без отрыва основных рабочих — слесарей от выполнения главных операций; определен комплект организационно-технической оснастки для размещения и хранения на рабочем месте инструментов, приспособлений, материалов и пр., а также для создания удобств рабочему при выполнении технологических операций и охраны труда; осуществлена рациональная планировка рабочих мест, избавляющая рабочих от лишних и утомительных трудовых движений и обеспечивающая удобную рабочую позу, рациональность трудового процесса и безопасность работы.

В целях экономии движений и устранения ненужных поисков предметы на рабочем месте делят на предметы постоянного и временного пользования, за которыми постоянно закреплены места хранения и расположения.

Исходными данными для разработки планировки цеха, участка, мастерской являются состав и габариты основного оборудования и организационно-технической оснастки рабочих мест, а также форм организации тру_-да и производства. Изучением вопросов рациональной организации трудового процесса занимается эргономика. Это сравнительно новая наука, изучающая функциональные возможности человека в трудовых процессах. Цель эргономики — создать наилучшие условия работы, при которых труд был бы высокопроизводительным и безопасным; обеспечивая рабочему необходимые удобства, сохранял его здоровье и высокую работоспособность.

Количество материалов или заготовок, находящихся на рабочем месте, деталей, собираемых в узлы, должно обеспечивать бесперебойную работу. Все материалы, заготовки и детали необходимо хранить в таре, на подставках или стеллажах. Расстояния от тары с заготовками и готовой продукцией и от оборудования (верстака) до рабочего должны быть такими, чтобы он мог использовать преимущественно движение рук. При этом учитывают, что при выполнении трудовых приемов, связанных с небольшими сопротивлениями усилию, особенно при выполнении работ, требующих повышенного внимания и точности, в работу включают мелкие звенья руки (кисть или даже одни пальцы). При выполнении приемов, связанных с условиями средней величины (4—5 кг) при небольших амплитудах, движение совершают за счет мышц плеча и предплечья и, наконец, при выполнении приемов, связанных со значительным усилием (6—8 кг), в движении принимает участие вся рука и даже корпус рабочего.

На рабочем месте должны находиться только те предметы, которые необходимы для выполнения данного задания. Предметы, которыми рабочий пользуется чаще, кладут ближе, в зоне досягаемости рук, ограниченной в горизонтальной плоскости дугами 1 (рис. 3.2,а). При работе сидя радиус дуги определяется движением согнутой в локтевом суставе руки, что составляет примерно 350 мм для каждой руки. Максимальная зона досягаемости составляет примерно 500 мм (она ограничивается дугами 2) и 600 мм с наклоном не более чем на 30° для работающего среднего роста — дугами 3 (см. рис. 3.2 ,а).

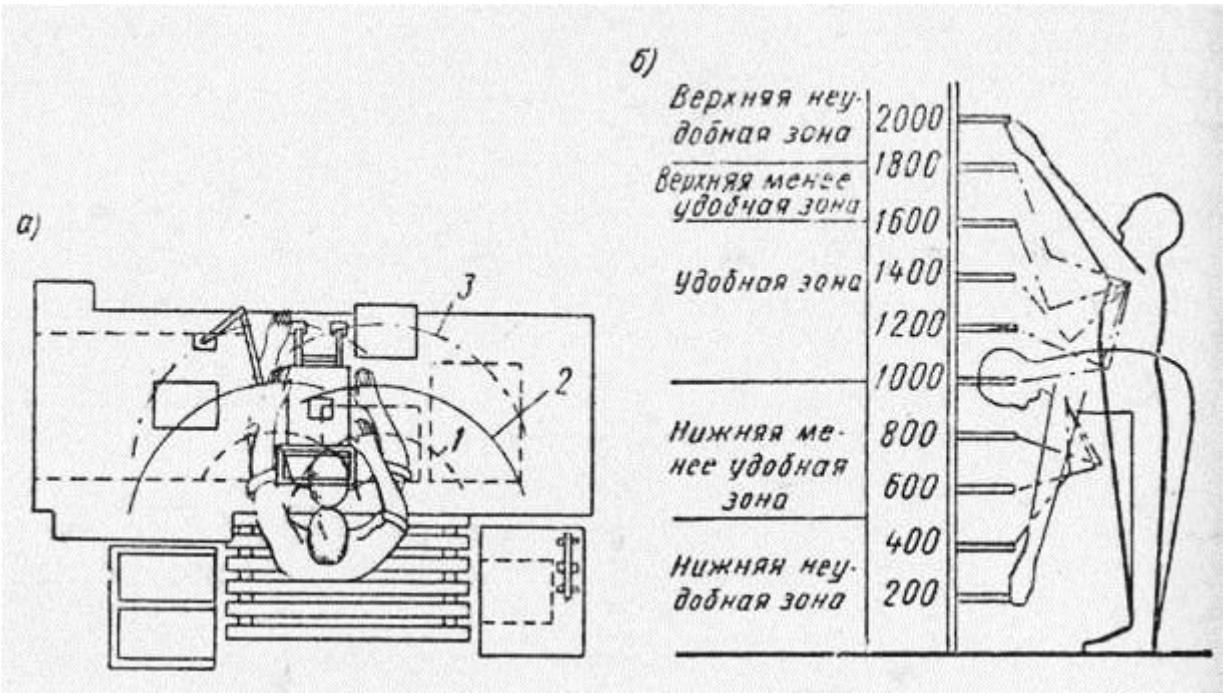

Рис. 3.2 Зоны досягаемости рук человека:
а — в горизонтальной плоскости при работе стоя и сидя; б—в вертикальной плоскости при работе стоя.

Расположение предметов далее указанных пределов повлечет за собой дополнительные и, следовательно, лишние движения, большие наклоны корпуса или лишние шаги. Все это вызовет лишнюю затрату рабочего времени, увеличит утомляемость рабочего и снизит производительность труда. Зоны досягаемости рук работающего стоя (в вертикальной плоскости)

приведены на рис. 1. Эти зоны дают возможность определить наиболее выгодное расположение всех предметов по отношению к росту работающего. Руководствуясь этими зонами, следует определять, на какой высоте от пола должны находиться материалы, заготовки, детали, приспособления, чтобы рабочему не приходилось низко наклоняться. Все предметы, которые приходится брать двумя руками, кладут прямо перед собой.

Рациональная планировка рабочих мест должна обеспечить выполнение коротких и менее утомительных движений рук, исключать широко применяемое при неправильных планировках перекладывание предметов (инструмента, заготовок, деталей и др.) из одной руки в другую. Все предметы, которые рабочий берет правой рукой, должны быть справа, а предметы, которые он берет левой рукой,— слева. Предметы, которыми пользуется рабочий в процессе труда, следует удобно расположить, а лишние предметы удалить с рабочего места. Заготовки должны быть подняты на высоту рук рабочего, чтобы он мог брать их не нагибаясь.

Для разработки рациональной планировки и организации рабочего места следует провести наблюдение за всеми трудовыми движениями работающего при выполнении слесарных операций. Выявив нерациональные трудовые движения и их причины, необходимо разработать проект рационального трудового процесса рабочего и соответственно этому составить планировку рабочего места, зафиксировав ее в карте организации рабочего места.

Поза работающего во время работы оказывает решающее влияние на его трудоспособность. Правильная рабочая поза обеспечивает сохранение продолжительной работоспособности. Известно, что работать можно стоя и сидя.

Изучение выявило, что наиболее утомительной является поза стоя, так как работающему приходится затрачивать значительную энергию на поддержание тела в вертикальном положении.

Необходимо иметь в виду, что любая поза человека является сложным координирующим процессом центральной нервной системы. При статическом удержании тела длительное время в одном и том же положении нервные клетки, управляющие соответствующими мышцами, все время возбуждены, и это вызывает рано наступающее утомление. Следует внимательно изучать условия работы каждой профессии рабочих в конкретных условиях производства и стремиться организовать, по возможности, работу сидя. Как показали исследования, производительность труда при этом увеличивается примерно на 10%, так как рабочие меньше утомляются. Однако работа в одном положении также приводит к утомлению, поэтому нужно устанавливать такой режим работы, при котором бы происходила смена рабочих поз в течение дня, т. е. работа стоя сменялась бы работой сидя и наоборот.

Исходя из требований НОТ на производстве, необходимо обеспечивать работающему удобные рабочие позы, используя для этого удобные сиденья, подножки-упоры, подъемно-винтовые стулья, стулья с подлокотниками и т. п. С рабочей позой тесно связаны физиологические процессы человека во время работы: дыхание, кровообращение, мускульные усилия и т. д. Рабочая поза оказывает влияние на точность и эффективность трудовых движений.

Научная организация труда на рабочем месте основывается на правильном режиме труда и отдыха, обеспечивающем поддержание высокой работоспособности человека и его здоровья. Исследования показали, что производительность труда в течение смены неодинакова. Работоспособность человека в течение смены делится на три этапа: первый этап — рабочий «входит» в работу, и постепенно растет его производительность; второй этап — период высокой производительности и третий этап — наступление усталости и ее нарастание. Установлено, что производительность труда в течение первых двух часов растет; высокий уровень ее держится около полутора часов и затем постепенно снижается, так как наступает утомление.

Для восстановления работоспособности работающих следует делать перерывы в зависимости от характера труда от 5 до 15 мин. как в первую, так и во вторую половину дня. Перерывы не должны быть особенно велики, однако они должны быть достаточными для восстановления психофизиологических функций организма. Перерывы полезно использовать для производственной гимнастики, а при особенно напряженной работе рабочие должны отдыхать в специальных комнатах отдыха.

Научная организация труда предполагает создание благоприятной производственной обстановки на рабочем месте. В комплекс элементов, создающих производственную обстановку, наряду с оргтехоснасткой входят способы окраски помещений и оборудования, состояние полов, оформление деталей производственных помещений, а также такие элементы гигиенических условий труда, как чистота, температура и влажность воздуха, уровень шума, гигиеническое, рациональное освещение и т. д. Производственная обстановка, окружая изо дня в день рабочего, оказывает на него большое влияние. Она может вызвать настроение подъема, активности, желание лучше и больше работать; она может также создавать настроение равнодушия, безразличия и даже уныния, пассивности, упадка и нежелания работать. Следовательно, нельзя недооценивать значения производственной обстановки, необходимо правильно использовать этот резерв улучшения качества работы и повышения производительности труд.

Вывод к третьей главе

В данной главе были рассмотрены: психофизиологические нагрузки на человека во время рабочего времени также была рассмотрена рациональная организация рабочего места

ЗАКЛЮЧЕНИЕ

Целью данной работы является создание мобильное приложение по планировке дня для платформы iOS. Приложение предназначено для пользователей желающие упорядочить свой рабочий день и увеличить свою производительность. В приложение должны входить функции добавления заметок, удаление и установление пользовательских настроек.

В рамках данной работы были получены следующие результаты:

- Приведена история телефонов и анализ истории.
- Рассмотрены современные мобильные операционные системы.
- Приведены типы мобильных приложений и их достоинства и недостатки.
- Приведены главные функции работы приложения.
- Разработана и реализована структурная схема базы данных
- Приведено подробное рукводсво пользователя
- Рассмотрена безопасность жизнедеятельности

СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ

1. Постановление Президента Республики Узбекистан «о мерах по дальнейшему внедрению и развитию современных информационно коммуникационных технологий» (2012г.)
2. http://technologies.uzreport.uz/news_r_104872.html - Статья «В Ташкенте обсуждены приоритетные направления совершенствования законодательной базы развития рынка программных продуктов»
3. Здзиарски Дж. iPhone SDK. Разработка приложений. — СПб: БХВ-Петербург, 2010. — 512 с.
4. Конвэй Дж., Хиллегасс А. Программирование под iOS. Для профессионалов. - СПб: Питер, 2013. — 608 с.
5. Марк Д., Натинг Д., Ламарш Д. Разработка приложений для iPhone, iPad и iPod Touch с использованием iOS SDK. - М.: ООО «И.Д. Вильямс», 2012. — 624 с.
6. Махер А. Программирование для iPhone. — М.: Эксмо, 2010. — 368 с.
7. Пейлон Д., Пейлон Т. Программируем для iPhone и iPad. 2-е изд. — СПб: Питер, 2012. — 624 с.
8. Ссылка на статью: типы мобильных приложений http://www.cmsmagazine.ru/library/items/moblile/native-vs-html5-vs-hybrid/
9. Реферат: Программные платформы современных смартфонов http://www.bestreferat.ru/referat-211733.html
10. Тайм менеджмент и организация рабочего времени http://www.infortech.ru/support/kb/files/time_management.html
11. Рациональная организация рабочего места и трудового процесса http://pereosnastka.ru/articles/ratsionalnaya-organizatsiya-rabochego-mesta-i-trudovogo-protsessa

12. http://www.znakcomplect.ru/novosti/example/2011-10-03-tematicheskie-stati-mery-bezopasnosti-pri-rabotax-po-modernizacii-kompyutera.phtml – Техника безопасности при работе на персональном компьютере. (20.04.2013)
13. http://www.lex.uz/Pages/GetAct.aspx?lact_id=1521663 – Закон Республики Узбекистан «о пожарной безопасности». (01.05.2013)

ПРИЛОЖЕНИЕ

//Класс главного окна расписаний

```
#import <UIKit/UIKit.h>
#import "MZDayPicker.h"
#import "PNote.h"
#import "PDay.h"
#import "PImage.h"
#import "LSMainCell.h"
#import "KGModal.h"
#import  <MapKit/MapKit.h>
@interface LSDayController : UIViewController <MKMapViewDelegate>
{
   PDay *currentDay;
   NSMutableArray *notes;
}
@property (weak, nonatomic) IBOutlet UITableView *tableView;
@property (weak, nonatomic) IBOutlet MZDayPicker *dayPicker;
@property (nonatomic,retain) NSManagedObjectContext *context;
@end
#define Pday @"PDay"
#define note @"PNote"
#import "LSDayController.h"
#import "LSAppDelegate.h"
#import "Plocation.h"
#import "LSSettingsStore.h"
#import "PContact.h"
@interface LSDayController () <MZDayPickerDelegate, MZDayPickerDataSource,
UITableViewDataSource, UITableViewDelegate>
{
   NSDate *Mydat;
}
@property (nonatomic,strong) NSMutableArray *tableData;
@property (nonatomic,strong) NSDateFormatter *dateFormatter;
@end
@implementation LSDayController
- (id)initWithNibName:(NSString *)nibNameOrNil bundle:(NSBundle *)nibBundleOrNil
{
   self = [super initWithNibName:nibNameOrNil bundle:nibBundleOrNil];
   if (self) {
      self.title = @"РцР»Р°РЅС‹";
      self.tabBarItem.image  = [UIImage imageNamed:@"calendar-25.png"];
      // Custom initialization
   }
   return self;
}
-(LSAppDelegate *) appDelegate
{
   return (LSAppDelegate*)[UIApplication sharedApplication].delegate;
}
- (void)viewDidLoad
{

   Mydat = [NSDate date];
   [super viewDidLoad];
```

```
    self.context = [self appDelegate].managedObjectContext;
    notes = [[NSMutableArray alloc] initWithCapacity:0];
    [[self navigationController] setNavigationBarHidden:true];

}
-(void)viewWillAppear:(BOOL)animated
{
    [self curweek];
    [self initDaywitDate:Mydat];
    [self getMageObject];
    [self.tableView  reloadData];


}
-(void)viewDidAppear:(BOOL)animated
{
    if (notes!= NULL || [notes count]>4) {
        int ind = 0;
        for (int i= 0 ;i<notes.count; i++) {
            PNote *not  = [notes objectAtIndex:i];
            if (not.noteTime >= [NSDate date]) {
                ind = i;

            }

        }
        if (ind != 0) {
            NSIndexPath *index = [NSIndexPath indexPathForRow:ind inSection:0];
            [self.tableView scrollToRowAtIndexPath:index atScrollPosition:UITableViewScrollPositionTop
animated:TRUE];
        }

    }

}
-(void)initDaywitDate:(NSDate*)dat
{
    NSFetchRequest *fetch = [[NSFetchRequest alloc]initWithEntityName:@"PDay"];
    NSArray *days = [self.context executeFetchRequest:fetch error:nil];
    NSDateFormatter *datef = [[NSDateFormatter alloc]init];
    datef.dateFormat = @"dd.MM.yyyy";
    NSString *str = [datef stringFromDate:dat];

    int count = 0;
    for (PDay*curDay in days) {

        NSString * dat =[datef stringFromDate:curDay.dateOfDay];
        if ([dat isEqualToString:str] ) {
            currentDay = curDay;
            count++;

        }
    }
    if (count == 0) {
```

```
        PDay *today =[NSEntityDescription insertNewObjectForEntityForName:Pday
inManagedObjectContext:self.context];
        today.dateOfDay = [datef dateFromString:str];
        NSError *savingErro=nil;
        if ([self.context save:&savingErro]) {
            NSLog(@"Save Done");
        }
        else{
            NSLog(@"Error");
        }
        days = [self.context executeFetchRequest:fetch error:nil];
        for (PDay*curDay in days) {
            NSString * dat =[datef stringFromDate:curDay.dateOfDay];
            if ([dat isEqualToString:str] ) {
                currentDay = curDay;

            }

        }

    }

    for (PDay*curDay in days) {
        NSString * dat =[datef stringFromDate:curDay.dateOfDay];
        NSLog(@"%@",dat);

    }

}
-(void) getMageObject
{
    [notes removeAllObjects];
    NSFetchRequest *fetch = [[NSFetchRequest alloc]initWithEntityName:note];
    NSSortDescriptor *sort = [[NSSortDescriptor alloc]initWithKey:@"noteTime" ascending:true];
    fetch.sortDescriptors = [NSArray arrayWithObject:sort];
    NSArray *temp = [self.context executeFetchRequest:fetch error:nil];
    for (int i = 0 ; i<temp.count; i++) {
        PNote * not = [temp objectAtIndex:i];
        if (not.toDay  == currentDay ) {
            [notes addObject:not];
        }
    }

}
-(void)curweek
{
     [[LSSettingsStore sharedInstance] load];

    self.tableData = [@[] mutableCopy];

    self.tableView.delegate = self;
    self.tableView.dataSource = self;

    self.dayPicker.delegate = self;
    self.dayPicker.dataSource = self;
```

```
    self.dayPicker.bottomBorderColor = [LSSettingsStore sharedInstance].slc;

    self.dayPicker.dayNameLabelFontSize = 12.0f;
    self.dayPicker.dayLabelFontSize = 18.0f;

    self.dateFormatter = [[NSDateFormatter alloc] init];
    [self.dateFormatter setDateFormat:@"EE"];
    //      self.dayPicker.month = 4;
    //      self.dayPicker.year = 2014;
    //      [self.dayPicker setActiveDaysFrom:1 toDay:30];
    //     [self.dayPicker setCurrentDay:24 animated:YES];


    //    [self.dayPicker setStartDate:[NSDate dateFromDay:22 month:4 year:2014] endDate:[NSDate
dateFromDay:26 month:4 year:2014]];
    //
    //    [self.dayPicker setCurrentDate:[NSDate dateFromDay:24 month:4 year:2014] animated:NO];

    self.tableView.frame = CGRectMake(0, self.dayPicker.frame.origin.y +
self.dayPicker.frame.size.height, self.tableView.frame.size.width, self.view.bounds.size.height-
self.dayPicker.frame.size.height);
    NSCalendar *cal = [NSCalendar currentCalendar];
    NSDate *curDate = [NSDate date];
    NSDateComponents *comp = [[NSDateComponents alloc]init];

    comp.day = -[[LSSettingsStore sharedInstance].visibleDays intValue]/2;
    NSDateComponents *comp2 = [[NSDateComponents alloc]init];
    comp2.day = [[LSSettingsStore sharedInstance].visibleDays intValue]/2;

    NSLog(@"%@",[cal dateByAddingComponents:comp toDate:curDate options:0]);
      [self.dayPicker setStartDate:[cal dateByAddingComponents:comp toDate:curDate options:0]
endDate:[cal dateByAddingComponents:comp2 toDate:curDate options:0]];
    [self.dayPicker setCurrentDate:[NSDate date] animated:NO];

}

- (NSString *)dayPicker:(MZDayPicker *)dayPicker titleForCellDayNameLabelInDay:(MZDay *)day
{
    return [self.dateFormatter stringFromDate:day.date];
}


- (void)dayPicker:(MZDayPicker *)dayPicker didSelectDay:(MZDay *)day
{
    NSLog(@"Did select day %@",day.date);
    Mydat = day.date;
    [self initDaywitDate:day.date];
    [self getMageObject];
    [self.tableView reloadData];
}

- (void)dayPicker:(MZDayPicker *)dayPicker willSelectDay:(MZDay *)day
{
    NSLog(@"Will select day %@",day.day);
```

```
}

#pragma mark - UITableViewDataSource

- (NSInteger)numberOfSectionsInTableView:(UITableView *)tableView
{
   return 1;
}

- (NSInteger)tableView:(UITableView *)tableView numberOfRowsInSection:(NSInteger)section
{
   return notes.count;
}

- (UITableViewCell *)tableView:(UITableView *)tableView cellForRowAtIndexPath:(NSIndexPath
*)indexPath
{
   static NSString *CellIdentifier = @"LSMainCell";
   LSMainCell *cell = (LSMainCell*)[tableView dequeueReusableCellWithIdentifier:CellIdentifier];
   if (cell ==nil) {
      NSArray *nib = [[NSBundle mainBundle] loadNibNamed:CellIdentifier owner:self options:nil];
      cell = [nib objectAtIndex:0];
   }
   PNote *tnote = [notes objectAtIndex:indexPath.row];
   if (tnote.toImage.count == 0 ) {
      cell.imageButton.hidden = true;
   }
   else
   {
      cell.imageButton.hidden = false;
      [cell.imageButton addTarget:self action:@selector(ShowImage:)
forControlEvents:UIControlEventTouchUpInside];
      [cell.imageButton setTag:indexPath.row+1];
   }
   if ([tnote.isNotify intValue] == 0) {
      cell.alarmButton.hidden = true;
   }
   else
   {
      cell.alarmButton.hidden = false;
   }
   if (tnote.contacts.count == 0 ) {
      cell.contactButton.hidden = true;
   }
   else
   {
      cell.contactButton.hidden = false;
      [cell.contactButton addTarget:self action:@selector(ShowContact:)
forControlEvents:UIControlEventTouchUpInside];
      [cell.contactButton setTag:indexPath.row+1];
   }
   if (tnote.toLocation.count == 0 ) {
      cell.loactionButton.hidden = true;
   }
   else
```

```
    {
        cell.loactionButton.hidden = false;
        [cell.loactionButton addTarget:self action:@selector(ShowLocation:)
forControlEvents:UIControlEventTouchUpInside];
        [cell.loactionButton setTag:indexPath.row+1];
    }
    if ([tnote.prior intValue] == 0) {cell.backgroundColor = [LSSettingsStore sharedInstance].npc;}
        else if ([tnote.prior intValue] == 1) {cell.backgroundColor = [LSSettingsStore sharedInstance].hpc;}
        else if([tnote.prior intValue] == 2){cell.backgroundColor = [LSSettingsStore sharedInstance].lpc;}


    cell.selectionStyle = UITableViewCellSelectionStyleNone;
    cell.nameLabel.text = tnote.noteName;
    NSDateFormatter *formatter = [NSDateFormatter new];
    formatter.dateFormat = @"HH:mm";

    cell.dateLabel.text = [formatter stringFromDate:tnote.noteTime];



    return cell;
}
- (BOOL)tableView:(UITableView *)tableView canEditRowAtIndexPath:(NSIndexPath *)indexPath
{
    // Return NO if you do not want the specified item to be editable.
    return YES;
}
- (NSString *)tableView:(UITableView *)tableView
titleForDeleteConfirmationButtonForRowAtIndexPath:(NSIndexPath *)indexPath
{
    return @"Удалить ";
}
- (void)tableView:(UITableView *)tableView
commitEditingStyle:(UITableViewCellEditingStyle)editingStyle forRowAtIndexPath:(NSIndexPath
*)indexPath
{
    if (editingStyle == UITableViewCellEditingStyleDelete) {
        [currentDay removeToNoteObject:[notes objectAtIndex:indexPath.row]];
        NSError *savingErro=nil;
        if ([self.context save:&savingErro]) {
            NSLog(@"Save remove Done");
        }
        else{
            NSLog(@"Error");
        }
        [notes removeObjectAtIndex:indexPath.row];
        [self.tableView reloadData];
       // [tableView deleteRowsAtIndexPaths:@[indexPath]
withRowAnimation:UITableViewRowAnimationFade];
    } else if (editingStyle == UITableViewCellEditingStyleInsert) {
        // Create a new instance of the appropriate class, insert it into the array, and add a new row to the
table view.
    }
}
```

```
-(void)ShowImage:(UIButton*) sender
{
   PNote *tnote = [notes objectAtIndex:sender.tag -1];
   NSArray *arrimg = [tnote.toImage allObjects];
   PImage *img = [arrimg objectAtIndex:0];
   UIImageView *imgv= [[ UIImageView alloc]initWithFrame:CGRectMake(10, 10, 290, 200)];
   imgv.image = img.image;
  [[KGModal sharedInstance]showWithContentView:imgv andAnimated:true];

}

-(void) ShowContact:(UIButton*) sender
{
   PNote *tnote = [notes objectAtIndex:sender.tag -1];
   NSArray *cont= [tnote.contacts allObjects];
   PContact *ct = [cont objectAtIndex:0];

   UIAlertView * alert = [[UIAlertView alloc] initWithTitle:@"Контактная информация "
message:ct.contact delegate:self cancelButtonTitle:@"OK" otherButtonTitles: nil];
   [alert show];
}

-(void) ShowLocation: (UIButton*) sender
{
   PNote* tnote= [notes objectAtIndex:sender.tag -1];
    NSArray *arrimg = [tnote.toLocation allObjects];
   Plocation *loc = [arrimg objectAtIndex:0];

   MKMapView *map = [[MKMapView alloc] initWithFrame:CGRectMake(10, 10, 290, 200)];
     [[KGModal sharedInstance]showWithContentView:[self mapView:map andLo:loc]
andAnimated:true];
}
-(MKMapView*) mapView:(MKMapView*)map andLo:(Plocation*)location
{
   map.delegate = self;

   CLLocationCoordinate2D loc;

   loc.latitude = [location.latitude doubleValue];
   loc.longitude=  [location.longtitude doubleValue];

   //    //41.3101792, 69.2996979/41.3161103, 69.2548943
   MKCoordinateRegion region= MKCoordinateRegionMakeWithDistance(loc, 2000, 2000);
   [map setRegion:[map regionThatFits:region]animated:true];
   MKPointAnnotation *tempAnnot=[[MKPointAnnotation alloc]init];

   MKPinAnnotationView *annotView=[[MKPinAnnotationView alloc]init];
   tempAnnot.coordinate=loc;
   annotView.annotation=tempAnnot;
    [map addAnnotation:annotView.annotation];


   return map;
}
```

```
- (MKAnnotationView *)mapView:(MKMapView *)mapView viewForAnnotation:(id
<MKAnnotation>)annotation
{
   static NSString *identifier = @"MyLocation";

   MKPinAnnotationView *annotationView = (MKPinAnnotationView *) [mapView
dequeueReusableAnnotationViewWithIdentifier:identifier];
   if (annotationView == nil)
   {

      annotationView = [[MKPinAnnotationView alloc] initWithAnnotation:annotation
reuseIdentifier:identifier];

   }
   else
   {
      annotationView.annotation = annotation;
   }
   return annotationView;

}
- (void)didReceiveMemoryWarning
{
   [super didReceiveMemoryWarning];
   // Dispose of any resources that can be recreated.
}

- (void)viewDidUnload {
   [self setTableView:nil];
   [self setDayPicker:nil];
   [super viewDidUnload];
}


@end
```

//Класс главного добавления заметок

```
#import <UIKit/UIKit.h>
#import "LSMainCell.h"
#import "LSChooseTypeViewController.h"
#import "CustomIOS7AlertView.h"
#import "LSImageCell.h"
#import "PDay.h"
#import "PNote.h"
#import "PImage.h"
//#import "LSContactCell.h"
#import "ContactCell.h"
#import "LSAppDelegate.h"
#import "LSLocationCell.h"
#import <MapKit/MapKit.h>
#import "ChooseOnMapViewController.h"
#import <AddressBook/AddressBook.h>
#import <AddressBookUI/AddressBookUI.h>

@interface LSAddNoteViewController : UIViewController <UITableViewDataSource,
UITableViewDelegate, UITextFieldDelegate, CustomIOS7AlertViewDelegate,
```

```
UIImagePickerControllerDelegate, UINavigationControllerDelegate,
MKMapViewDelegate,UIPickerViewDataSource,UIPickerViewDelegate,ABPeoplePickerNavigationCon
trollerDelegate,UIActionSheetDelegate>
{
   int cellTag;
   UIDatePicker *datePicker;
   NSMutableDictionary *pictures;
   PDay *currentDay;
   NSMutableArray *ModelArr;
   UIPickerView *pickPrior;
   CustomIOS7AlertView *pAlert;

}
@property (weak, nonatomic) IBOutlet UITextField *nameField;
@property (weak, nonatomic) IBOutlet UISwitch *notifSwitcher;
- (IBAction)addClick:(id)sender;
@property (weak, nonatomic) IBOutlet UITableView *tableView;
- (IBAction)dateChoose:(id)sender;
@property (weak, nonatomic) IBOutlet UILabel *dateLabel;
@property (nonatomic, retain) NSManagedObjectContext *context;

@property (nonatomic,retain) NSMutableArray * types;
- (IBAction)ChoosePriop:(id)sender;
@property (weak, nonatomic) IBOutlet UILabel *priorlabel;
@property (nonatomic, strong) ABPeoplePickerNavigationController *addressBookController;
@property (strong, nonatomic) IBOutlet UIView *contView;

- (IBAction)TypesC:(id)sender;
@end

#import "LSAddNoteViewController.h"
#import "Plocation.h"
#import "LSLocationClass.h"
#import "PVoice.h"
#import "PContact.h"
#import "WToast.h"
#import "CMPopTipView.h"

@interface LSAddNoteViewController () <CMPopTipViewDelegate>

{
   MKMapView *mapViewP;
   LSLocationClass * cord;
   NSArray *forPick;
   NSNumber * prior;
   NSString * contact;
   CMPopTipView *popTipView;
}
@property (nonatomic, strong) NSMutableArray *arrContactsData;

@end

@implementation LSAddNoteViewController

- (id)initWithNibName:(NSString *)nibNameOrNil bundle:(NSBundle *)nibBundleOrNil
```

```
{
   self = [super initWithNibName:nibNameOrNil bundle:nibBundleOrNil];
   if (self) {
      self.title = @"P”PsP±P°PIPëC‚CЊ";
      self.tabBarItem.image  = [UIImage imageNamed:@"plus-25.png"];
      // Custom initialization
   }
   return self;
}


- (void)viewDidLoad
{
   [[NSNotificationCenter defaultCenter] addObserver:self selector:@selector(handleNoti:)
name:@"CordinatesNotif" object:nil];
   cord  = [LSLocationClass new];
   ModelArr = [[NSMutableArray alloc] initWithCapacity:0];

   [super viewDidLoad];
   [self resetData];
   self.context = [self appDelegate].managedObjectContext;
   [self initInter];
   [self initsome];
   [self initDaywitDate:[NSDate date]];
   forPick = @[@"PћP±C‹C‡PSC‹P№",@"P’C‹CЃPsPєPёP№", @"PќPёP·PєPёP№"];
   prior = [[NSNumber alloc] initWithInt:0];
   // Do any additional setup after loading the view from its nib.
}
-(void) resetData
{
   self.nameField.text = @"";
   NSDateFormatter *formatter = [NSDateFormatter new];
   formatter.dateFormat = @"dd.MM.yyyy hh:mm";
   self.dateLabel.text = [formatter stringFromDate:[NSDate date]];
   self.priorlabel.text = @"PћP±C‹C‡PSC‹P№";
   self.notifSwitcher.on = true;
   [self.types removeAllObjects];
   [ModelArr removeAllObjects];
   [self.tableView reloadData];


}
-(void)viewWillAppear:(BOOL)animated
{
   [self.tableView reloadData];
}

-(void) handleNoti:(NSNotification*)info
{
   cord = [info object];

}
-(void)initDaywitDate:(NSDate*)dat
{
   NSFetchRequest *fetch = [[NSFetchRequest alloc]initWithEntityName:@"PDay"];
```

```
    NSArray *days = [self.context executeFetchRequest:fetch error:nil];
    NSDateFormatter *datef = [[NSDateFormatter alloc]init];
    datef.dateFormat = @"dd.MM.yyyy";
    NSString *str = [datef stringFromDate:dat];

    int count = 0;
    for (PDay*curDay in days) {

        NSString * dat =[datef stringFromDate:curDay.dateOfDay];
        if ([dat isEqualToString:str] ) {
            currentDay = curDay;
            count++;

        }
    }
    if (count == 0) {
        PDay *today =[NSEntityDescription insertNewObjectForEntityForName:@"PDay"
inManagedObjectContext:self.context];
        today.dateOfDay = [datef dateFromString:str];
        NSError *savingErro=nil;
        if ([self.context save:&savingErro]) {
            NSLog(@"Save Done");
        }
        else{
            NSLog(@"Error");
        }
        days = [self.context executeFetchRequest:fetch error:nil];
        for (PDay*curDay in days) {
            NSString * dat =[datef stringFromDate:curDay.dateOfDay];
            if ([dat isEqualToString:str] ) {
                currentDay = curDay;

            }

        }

    }


    for (PDay*curDay in days) {

        NSString * dat =[datef stringFromDate:curDay.dateOfDay];
        NSLog(@"%@",dat);

    }

}

-(void)initsome
{
    pictures = [[NSMutableDictionary alloc]initWithCapacity:0];
}
-(void)initInter
{
    self.navigationItem.title = @"Р”РѕР±Р°РІРёС‚СЊ";
```

```
        UIBarButtonItem *addItem = [[UIBarButtonItem
alloc]initWithBarButtonSystemItem:UIBarButtonSystemItemAdd target:self
action:@selector(addType:)];
    self.navigationItem.rightBarButtonItem = addItem;
    NSDateFormatter *formatter = [NSDateFormatter new];
    formatter.dateFormat = @"dd.MM.yyyy hh:mm";
    self.dateLabel.text = [formatter stringFromDate:[NSDate date]];
    self.types = [[NSMutableArray alloc]initWithCapacity:0];
    self.nameField.delegate =self;
}


-(BOOL)peoplePickerNavigationController:(ABPeoplePickerNavigationController *)peoplePicker
shouldContinueAfterSelectingPerson:(ABRecordRef)person property:(ABPropertyID)property
identifier:(ABMultiValueIdentifier)identifier{
   contact = @"";

    ABMultiValueRef phoneNumbers = ABRecordCopyValue(person, property);

    contact = (__bridge_transfer NSString*) ABMultiValueCopyValueAtIndex(phoneNumbers, identifier);
      ContactCell *cell =(ContactCell*) [self.tableView viewWithTag:cellTag];
    cell.typeTextfield.text = contact;
    [_addressBookController dismissViewControllerAnimated:YES completion:nil];
    return TRUE;

}


-(void)peoplePickerNavigationControllerDidCancel:(ABPeoplePickerNavigationController
*)peoplePicker{
    [_addressBookController dismissViewControllerAnimated:YES completion:nil];
}
- (NSInteger)numberOfSectionsInTableView:(UITableView *)tableView
{
    return 1;
}
- (NSInteger)tableView:(UITableView *)tableView numberOfRowsInSection:(NSInteger)section
{

    // Return the number of rows in the section.
    return self.types.count;
}

- (UITableViewCell *)tableView:(UITableView *)tableView cellForRowAtIndexPath:(NSIndexPath
*)indexPath
{
    if ([[_types objectAtIndex:indexPath.row]intValue] == 1) {
        static NSString *CellIdentifier = @"LSImageCell";
        LSImageCell *cell = (LSImageCell*)[tableView dequeueReusableCellWithIdentifier:CellIdentifier];
        if (cell ==nil) {
            NSArray *nib = [[NSBundle mainBundle] loadNibNamed:CellIdentifier owner:self options:nil];
            cell = [nib objectAtIndex:0];
        }

        // Configure the cell...
```

```
        [cell.buttonPick addTarget:self action:@selector(pickImage:)
forControlEvents:UIControlEventTouchUpInside];
        [cell.buttonPick setTag:indexPath.row+1];
        [cell setTag:indexPath.row+1];
        return cell;

    }
   else if ([[_types objectAtIndex:indexPath.row]intValue] == 2) {
        static NSString *CellIdentifier = @"LSLocationCell";
        LSLocationCell *cell = (LSLocationCell*)[tableView
dequeueReusableCellWithIdentifier:CellIdentifier];
        if (cell ==nil) {
            NSArray *nib = [[NSBundle mainBundle] loadNibNamed:CellIdentifier owner:self options:nil];
            cell = [nib objectAtIndex:0];
        }

        // Configure the cell...

       UITapGestureRecognizer *rec = [[UITapGestureRecognizer alloc]initWithTarget:self
action:@selector(handeleTap:)];

        [cell.MapView addGestureRecognizer:rec];
        cell.MapView = [self mapView:cell.MapView];
    //   mapView = cell.MapView;
        return cell;

    }
   else if ([[_types objectAtIndex:indexPath.row]intValue] == 3) {
        static NSString *CellIdentifier = @"ContactCell";
        ContactCell*cell = (ContactCell*)[tableView dequeueReusableCellWithIdentifier:CellIdentifier];
        if (cell ==nil) {
            NSArray *nib = [[NSBundle mainBundle] loadNibNamed:CellIdentifier owner:self options:nil];
            cell = [nib objectAtIndex:0];
        }

        [cell.typeTextfield setTag:indexPath.row +1];
        [cell.typeLabel setTag:indexPath.row +1];
        [cell setTag:indexPath.row +1];
        [cell.addButtom setTag:indexPath.row +1];
        [cell.addButtom addTarget:self action:@selector(contactChoose:)
forControlEvents:UIControlEventTouchUpInside];
        return cell;
   }
    return [UITableViewCell new];

}
- (BOOL)tableView:(UITableView *)tableView canEditRowAtIndexPath:(NSIndexPath *)indexPath
{
    // Return NO if you do not want the specified item to be editable.
    return YES;
}
- (NSString *)tableView:(UITableView *)tableView
titleForDeleteConfirmationButtonForRowAtIndexPath:(NSIndexPath *)indexPath
{
    return @"Удалить";
```

```
}
- (void)tableView:(UITableView *)tableView
commitEditingStyle:(UITableViewCellEditingStyle)editingStyle forRowAtIndexPath:(NSIndexPath
*)indexPath
{
   if (editingStyle == UITableViewCellEditingStyleDelete) {
   [self.types removeObjectAtIndex:indexPath.row];
      [ModelArr removeObjectAtIndex:indexPath.row];
     [tableView deleteRowsAtIndexPaths:@[indexPath]
withRowAnimation:UITableViewRowAnimationFade];
   } else if (editingStyle == UITableViewCellEditingStyleInsert) {
      // Create a new instance of the appropriate class, insert it into the array, and add a new row to the
table view.
   }
}

-(MKMapView*) mapView:(MKMapView*)map
{
   map.delegate = self;

   CLLocationCoordinate2D loc;

   loc.latitude= 41.3101792;
   loc.longitude= 69.2996979;
   if ([cord.lat doubleValue] != 0.0 && [cord.lon doubleValue]!= 0.0) {
      loc.latitude = [cord.lat doubleValue];
      loc.longitude = [cord.lon doubleValue];
   }
   //    //41.3101792, 69.2996979/41.3161103, 69.2548943
   MKCoordinateRegion region= MKCoordinateRegionMakeWithDistance(loc, 2000, 2000);
   [map setRegion:[map regionThatFits:region]animated:true];
   MKPointAnnotation *tempAnnot=[[MKPointAnnotation alloc]init];

   MKPinAnnotationView *annotView=[[MKPinAnnotationView alloc]init];
   tempAnnot.coordinate=loc;
   annotView.annotation=tempAnnot;

   [annotView setTag:1];

  if ([cord.lat doubleValue] != 0.0 && [cord.lon doubleValue]!= 0.0) {
      [map addAnnotation:annotView.annotation];
   }

   return map;
}
- (void)didReceiveMemoryWarning
{
   [super didReceiveMemoryWarning];
   // Dispose of any resources that can be recreated.
}

-(void) contactChoose:(UIButton*) sender
{
    cellTag = sender.tag;
   _addressBookController = [[ABPeoplePickerNavigationController alloc] init];
```

```
    [_addressBookController setPeoplePickerDelegate:self];
    [self presentViewController:_addressBookController animated:YES completion:nil];
}
-(void)handeleTap:(UITapGestureRecognizer*) rec
{
    ChooseOnMapViewController *viewCont = [ChooseOnMapViewController new];
    [self .navigationController pushViewController:viewCont animated:true];

-(void) addType:(id) sender
{
        popTipView = [[CMPopTipView alloc] initWithCustomView:self.contView];
         popTipView.delegate = self;
         popTipView.dismissTapAnywhere = YES;
        popTipView.backgroundColor = [UIColor colorWithRed:244.0f green:244.0f blue:244.0f alpha:1.0];
         UIBarButtonItem *barButtonItem = (UIBarButtonItem *)sender;
         [popTipView presentPointingAtBarButtonItem:barButtonItem animated:YES];
}
- (void)popTipViewWasDismissedByUser:(CMPopTipView *)popTipView
{
            //[popTipView dismissAnimated:TRUE];
}

- (IBAction)addClick:(id)sender {
    if ([self.nameField.text length] == 0) {
        [WToast showWithText:@"PµCŕCЃC‚PsPµ PїPsP»Pµ PёPjPµPSPё"];
        return;
    }

    if (datePicker != Nil) {
        [self initDaywitDate:datePicker.date];
    }
    NSManagedObjectContext *context = [self appDelegate].managedObjectContext;
   PNote * Mynote = [NSEntityDescription insertNewObjectForEntityForName:@"PNote"
inManagedObjectContext:context];
    Mynote.noteName = self.nameField.text;
    Mynote.isNotify = [NSNumber numberWithBool:self.notifSwitcher.on];
    if (datePicker == Nil) {
        Mynote.noteTime = [NSDate date];
    }
    else
    {
        Mynote.noteTime = datePicker.date;
    }
    Mynote.prior = prior;
    Mynote.toDay = currentDay;
    for (NSManagedObject *obj in ModelArr) {
        if ([obj isKindOfClass:[PImage class]]) {
            PImage *image =(PImage*) obj;
            image.image = [pictures objectForKey:[NSString stringWithFormat:@"%d",cellTag]];
            [Mynote addToImageObject:image];
        }
        if ([obj isKindOfClass:[Plocation class]]) {
            Plocation *loc =(Plocation*) obj;
            loc.latitude = cord.lat;
            loc.longtitude = cord.lon;
```

```
            [Mynote addToLocationObject:loc];
        }
        if ([obj isKindOfClass:[PContact class]]) {
            PContact *loc =(PContact*) obj;
            loc.contact = contact;
            [Mynote addContactsObject:loc];
        }
    }

    NSError *savingErro=nil;
    if ([context save:&savingErro]) {
        if (Mynote.isNotify) {
            UILocalNotification* localNotification = [[UILocalNotification alloc] init];
            localNotification.fireDate = Mynote.noteTime;
            localNotification.alertBody = Mynote.noteName;
            localNotification.alertAction = @"PќP°PїPsPjPёPSP°PSPёPµ";
            localNotification.timeZone = [NSTimeZone defaultTimeZone];
            localNotification.applicationIconBadgeNumber = [[UIApplication sharedApplication]
applicationIconBadgeNumber] + 1;
            [[UIApplication sharedApplication] scheduleLocalNotification:localNotification];

        }

        NSLog(@"Save Done");
        [WToast showWithText:@"P”PsP±P°PIP»PµPSPs CґCЃPїPµC€PSPs"];

    }
    else{
        NSLog(@"Error");
    }

    [self resetData];


}
- (IBAction)dateChoose:(id)sender {
    datePicker = [[UIDatePicker alloc]initWithFrame:CGRectMake(0, 0, 300, 150)];
    CustomIOS7AlertView *alert = [CustomIOS7AlertView new];
    alert.containerView = datePicker;
    alert.delegate = self;
    alert.buttonTitles = [NSArray arrayWithObject:@"P“PsC‚PsPIPs"];
    [alert show];

}
-(void)pickImage:(UIButton*) sender
{
    cellTag = sender.tag;
    UIActionSheet *act =[[UIActionSheet alloc]initWithTitle:@"P’C‹P±PµCЂPёC‚Pµ
PёP·PsP±CЂP°P¶PµPSPёPµ" delegate:self cancelButtonTitle:@"PћC‚PjPµPSP°"
destructiveButtonTitle:@"PЎPrPµP»P°C‚CЊ C„PsC‚PsPiCЂP°C„PёCЋ"
otherButtonTitles:@"P“P°P»PµCЂPµCЏ", nil];
    act.delegate =self;

    act.actionSheetStyle=UIActionSheetStyleBlackTranslucent;
    [act showInView:self.view];
```

```
-(void)actionSheet:(UIActionSheet *)actionSheet clickedButtonAtIndex:(NSInteger)buttonIndex
{
   UIImagePickerController *picker = [[UIImagePickerController alloc]init];
   picker.delegate=self;
   if (buttonIndex==1) {
      if ([UIImagePickerController
isSourceTypeAvailable:UIImagePickerControllerSourceTypePhotoLibrary]) {
         picker.sourceType=UIImagePickerControllerSourceTypePhotoLibrary;
         [self presentViewController:picker animated:true completion:nil];
         //NSLog(@"dasda");
      }
   }
   if (buttonIndex==0) {
      if ([UIImagePickerController isSourceTypeAvailable:UIImagePickerControllerSourceTypeCamera])
      {
         picker.sourceType=UIImagePickerControllerSourceTypeCamera;
         [self presentViewController:picker animated:true completion:nil];

      }
   }
}
- (void)imagePickerController:(UIImagePickerController *)picker
didFinishPickingMediaWithInfo:(NSDictionary *)info
{
   LSImageCell *cell =(LSImageCell*) [self.tableView viewWithTag:cellTag];
   cell.image.image = [info objectForKey:UIImagePickerControllerOriginalImage];
   [pictures setObject:[info objectForKey:UIImagePickerControllerOriginalImage] forKey:[NSString
stringWithFormat:@"%d",cellTag]];
   //[(UIImageView*)[self.view viewWithTag:12] setImage:info[UIImagePickerControllerEditedImage]
];

   [self.navigationController dismissViewControllerAnimated:true completion:nil];

}
- (void)imagePickerControllerDidCancel:(UIImagePickerController *)picker
{
   [self.navigationController dismissViewControllerAnimated:true completion:nil];
}


-(void)customIOS7dialogButtonTouchUpInside:(id)alertView
clickedButtonAtIndex:(NSInteger)buttonIndex
{
   if (alertView == pAlert) {
      [alertView close];
   }
   else
   {
   NSDateFormatter *formatter = [NSDateFormatter new];
   formatter.dateFormat = @"dd.MM.yyyy hh:mm";
   self.dateLabel.text = [formatter stringFromDate:datePicker.date];
   [alertView close];
   }
}
```

```
-(LSAppDelegate *) appDelegate
{
   return (LSAppDelegate*)[UIApplication sharedApplication].delegate;
}
-(BOOL)textFieldShouldReturn:(UITextField *)textField
{
   [self.nameField resignFirstResponder];
   return true;
}
- (MKAnnotationView *)mapView:(MKMapView *)mapView viewForAnnotation:(id
<MKAnnotation>)annotation
{
   MKPinAnnotationView *annotationView = (MKPinAnnotationView *) [mapView
dequeueReusableAnnotationViewWithIdentifier:identifier];
   if (annotationView == nil)
   {

      annotationView = [[MKPinAnnotationView alloc] initWithAnnotation:annotation
reuseIdentifier:identifier];

      [UIButton buttonWithType:UITableViewCellAccessoryDetailButton];
   }
   else
   {
      annotationView.annotation = annotation;
   }
   return annotationView;
}
- (NSInteger)pickerView:(UIPickerView *)pickerView
numberOfRowsInComponent:(NSInteger)component
{
   return forPick.count;
}
- (NSString *)pickerView:(UIPickerView *)pickerView titleForRow:(NSInteger)row
forComponent:(NSInteger)component
{
   return [forPick objectAtIndex:row];
}
- (void)pickerView:(UIPickerView *)pickerView didSelectRow:(NSInteger)row
inComponent:(NSInteger)component
{
   prior = [NSNumber numberWithInteger:row];
   self.priorlabel.text = [forPick objectAtIndex:row];
}
-(void) dealloc
{
   [[NSNotificationCenter defaultCenter] removeObserver:self];
}

- (IBAction)ChoosePriop:(id)sender {
   pAlert  = [CustomIOS7AlertView new];
   pAlert.buttonTitles = @[@"P’C‹P±CЂP°C,CЊ"];
   pickPrior = [[UIPickerView alloc]initWithFrame:CGRectMake(0, 0, 300, 150)];
   pickPrior.delegate = self;
   pAlert.containerView = pickPrior;
```

```
    [pAlert show];
}
- (IBAction)TypesC:(UIButton*)sender {
    int Chek = 0;
    for (int i = 0; i<self.types.count; i++) {
        if ([[self.types objectAtIndex:i]intValue] == sender.tag) {
            Chek++;

        }
    }
    if (Chek == 0) {
        [self.types addObject:[NSNumber numberWithInt:sender.tag]];
        NSManagedObjectContext *context = [self appDelegate].managedObjectContext;
        if (sender.tag -1 == 0) {
            PImage *image = [NSEntityDescription insertNewObjectForEntityForName:@"PImage"
inManagedObjectContext:context];
            [ModelArr addObject:image];


        }
        else if (sender.tag -1 == 1)
        {
            Plocation *location = [NSEntityDescription insertNewObjectForEntityForName:@"Plocation"
inManagedObjectContext:context];
            [ModelArr addObject:location];
        }
        else if (sender.tag -1 == 2)
        {
            PContact *cont = [NSEntityDescription insertNewObjectForEntityForName:@"PContact"
inManagedObjectContext:context];
            [ModelArr addObject:cont];

        }
        else if (sender.tag -1 == 3)
        {
            PVoice *voice = [NSEntityDescription insertNewObjectForEntityForName:@"PVoice"
inManagedObjectContext:context];
            [ModelArr addObject:voice];

        }
    }
    [self.tableView reloadData];
    [popTipView dismissAnimated:TRUE];
}
@end
```